REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	3.Sep.03	DISSERTATION

4. TITLE AND SUBTITLE
"A STUDY OF MLFMA FOR LARGE-SCALE SCATTERING PROBLEMS"

5. FUNDING NUMBERS

6. AUTHOR(S)
CAPT HASTRITER MICHAEL L

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
UNIVERSITY OF ILLINOIS AT URBANA

8. PERFORMING ORGANIZATION REPORT NUMBER
CI02-1271

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

14. SUBJECT TERMS	15. NUMBER OF PAGES	16. PRICE CODE
	156	

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

A STUDY OF MLFMA FOR LARGE-SCALE SCATTERING PROBLEMS

BY

MICHAEL LARKIN HASTRITER

B.S., Brigham Young University, 1993
M.S., Air Force Institute of Technology, 1997

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctorate of Philosophy in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2003

Urbana, Illinois

CERTIFICATE OF COMMITTEE APPROVAL

University of Illinois at Urbana-Champaign
Graduate College

June 19, 2003

We hereby recommend that the thesis by:

MICHAEL LARKIN HASTRITER

Entitled:

A STUDY OF MLFMA FOR LARGE-SCALE SCATTERING PROBLEMS

Be accepted in partial fulfillment of the requirements for the degree of:

Doctor of Philosophy

Signatures:

W. C. Chew
Director of Research

William R. Perkins
Head of Department
for R. E. Blahut

Committee on Final Examination*

W. C. Chew
Chairperson

Committee Member

Committee Member

Committee Member

Committee Member

Committee Member

* Required for doctoral degree but not for master's degree

ABSTRACT

This research is centered in computational electromagnetics with a focus on solving large-scale problems accurately in a timely fashion using first principle physics. Error control of the translation operator in 3-D is shown. A parallel implementation of the multilevel fast multipole algorithm (MLFMA) was studied as far as parallel efficiency and scaling. The large-scale scattering program (LSSP), based on the ScaleME library, was used to solve ultra-large-scale problems including a 200λ sphere with 20 million unknowns. As these large-scale problems were solved, techniques were developed to accurately estimate the memory requirements. Careful memory management is needed in order to solve these massive problems. The study of MLFMA in large-scale problems revealed significant errors that stemmed from inconsistencies in constants used by different parts of the algorithm. These were fixed to produce the most accurate data possible for large-scale surface scattering problems. Data was calculated on a missile-like target using both high frequency methods and MLFMA. This data was compared and analyzed to determine possible strategies to increase data acquisition speed and accuracy through multiple computation method hybridization.

The views expressed in this article are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U.S. Government

PREFACE

This preface may seem verbose and lengthy; however, such verbosity mirrors the details and the long challenges involved in working towards a doctorate degree. My first acknowledgment goes to my adviser, Weng Cho Chew and my committee made up of Andreas Cangellaris, Jian-Ming Jin, and Shung-Wu (Andy) Lee. They are all extraordinary professors. I am extremely grateful for the efforts of Dr. Sanjay Velamparambil, who authored the large-scale scattering program, which facilitated my research into parallel MLFMA. At home, a loving and wonderful wife who patiently endured the Ph.D. program deserves great recognition. More people have helped me and I recognize their contributions. I will attempt to discuss in detail those contributors who made possible my success. I strongly believe that the God of the universe, our Heavenly Father, played a distinct role in helping me during the research, classes, and challenges that I faced along the way. I know that He is keenly aware of the challenges that we all face, and He is willing to help us if we seek His help in our lives.

At the end of 1997, I finished a master's degree at the Air Force Institute of Technology (AFIT) under the direction of Dr. Peter Collins. He and other faculty suggested that I stay at AFIT to pursue a Ph.D. in electromagnetics. After carefully weighing my options and the possibility, at the time, of AFIT closing, I decided to exit the academic world for a season. I thought my exit would be permanent, but Major Collins invited me to interview for his faculty position as the Air Force was going to move him. Again, I declined the invitation because of careful consideration of activities at both work and church. The timing was not right, and the painful experience of education had not sufficiently faded from my memory. At last, after nearly two years, the position opened again and Collins was able to convince me that I should put it at the top of my list for my next job in the Air Force. I made it my second choice under the possibility of going to test pilot school (TPS). When I was medically disqualified for TPS based on a headache I had in 1995, the Ph.D. became my first choice. I interviewed in February 2000 and by June, I had orders to go to school. I am thankful for that headache and for the persistence of Pete Collins, now a lieutenant colonel in the Air Force, in recruiting me into the AFIT civilian institution program.

In the application process, Colonel Richard Stotts, then commander of the Air Force Information Warfare Center, brought me into his office to counsel me on the pros and cons of pursuing a Ph.D. in the Air Force. His support allowed me to successfully acquire the position after I convinced him that I was sincerely interested in the additional education. He told me that I was sacrificing a great potential career in the Air Force, but that a Ph.D. on the outside could be very valuable. I did not believe his opinion that my career would be hurt; however, I have taken his advice to 'finish on time' very seriously. To help my career, he supported me with a Meritorious Service Medal, which came at a critical time during my first semester at Illinois when I was feeling overwhelmed and depressed.

This feeling of inadequacy was mitigated by a blessing I received at the hands of Bishop Steve Newman. In this blessing, I received the promise of successful completion of the Ph.D. program and a promise that I would be healthy even when others around me were sick. I testify that this has been the case and that I have received many miracles of health and inspiration along this journey. I feel like the windows of heaven have been opened unto me. At one point, I caught a ride home with Paul Callister, a friend from church, and he listened to me share my struggles in the initial stages of completing the Ph.D. coursework and the pending qualifying exams. At least once, I had thought about jumping out of the window, but such thoughts do not have a divine source and when one weighs the value of life, it is easy to dismiss such temptations. Paul suggested that I might be in the wrong field and that success does not always follow hard work. This was an interesting concept to ponder, but I concluded that I was in the correct field and God could help me overcome any obstacles along the way.

In the first semester at the University of Illinois, I took Professor Chew's class on waves and fields in inhomogeneous media. I was missing both of the pre-requisites as we soon discovered. Chew helped me to succeed when failure seemed imminent. I am not the first one to walk out of his office with hope after entering with dismay. In this course, he could easily perceive when I was understanding and, more often, when I was puzzled. He regularly had to clarify instruction for my benefit. He is a master teacher and a great example of excellence in both teaching and research. His advice about coursework and related decisions was helpful. I was initially hesitant to put advanced math courses into my education plan, but he just said that we would talk about it later. Later, when the decisions

were important, he helped me to have the courage to enhance my mathematical maturity, an essential part of a successful Ph.D. program.

During the first semester, Dr. Chew also helped me get involved with a research project. On the research team, we played an advisory role where my Air Force signatures experience and contacts were useful. It was during this first semester that I also met a choice friend, Eric Dunn, who helped me through Chew's challenging class. Eric and I shared five classes together and another course, which he already took, he helped me to learn. Beginning at the end of my first semester, we launched an intensive plan to study for the qualifying exams. I put up a 4' x 8' whiteboard in my home and we spent many hours starting early each Saturday morning to prepare for this huge milestone. We also spent two regular weekday afternoons in group study for the qualifying exams up at Everitt Laboratory. Probably half of my time studying for this rite-of-passage was with Eric. Although I also received help from Dr. Bin Hu, Dr. Chris Pan, Yunhui Chu, Dr. Mark Skouson, and others, Eric was instrumental in helping me to surpass this milestone. I have felt indebted to him for helping me. We have collaborated on many things, and I predict that he will become one of the great teachers of the century.

Dr. Bin Hu helped me prepare for the qualifying exams. He always told me that I could make it. He correctly predicted, "You will pass; don't worry." I often sent e-mails to Bin Hu at 2:00 in the morning or later, and he would always send back an immediate response. Often his response was the key to finishing a homework assignment before sunrise. I was amazed at his dedication and diligence as a student. Dr. Siyuan Chen helped me avoid several unnecessary classes so that I could focus on classes to help me prepare for the qualifying exams. It was tough to see these two guys graduate and leave the group.

Once I passed the qualifying exams, several people told me that I was 75% finished with the Ph.D. degree. They were wrong. Passing the departmental exam was a monumental task that seemed impossible; however, it was a small peak compared to the research requirements. In fact, after passing the exam, the feeling of relief and happiness soon vanished as I began to glimpse what lay ahead. The night that the faculty met to decide our fates, Professor Chew called me at home sometime after 9:00 and congratulated me on passing the qual. I remember yelling with excitement and then apologizing for yelling into the telephone. I was grateful that he called me before the official release to put my mind at ease. He knew I

worked hard and kept my ego in check by hinting that I barely cleared the bar. I did my best and the qual gave me increased confidence and skills to begin my research.

At the end of the second semester, Chew had assigned me a project in his class on waveguides that turned into some research that lasted a good portion of the summer. To finish his class, I had to turn in the project, but it expanded into more work in the summer. The reason I mention this particular project is because it involved some major challenges that Professor Chew helped me to overcome. He carefully helped me formulate the problem. This project stretched me and at the end of the semester, I desperately needed his help. Shortly before the project was due, I was frantically trying to find out why my Matlab code was not working. I called him at his home late at night and he had just gotten home and had to go eat some dinner. He told me to call him back in an hour if I still needed help. I called him back and he invited me over to his house where he helped me put in the final piece of the puzzle to get things to work. Sacrifices like these have a big impact on students, and I will remember to assist my future students in the same way.

I have been fortunate to have great support from many teachers in my lifetime. Another example of such dedication is with Dr. Andy Lee. During my master's thesis, he helped me several times late at night. He was a major reason I chose to come to Illinois for a Ph.D. I was happy when he offered to help me on my Ph.D. and to serve on my committee. His work on Xpatch and its development provided key tools for me to complete my degree. While I mention Dr. Lee, I should also mention Dennis Andersh, who was my first supervisor and mentor on active duty in the Air Force. He and the Demaco division of Science Applications International Corporation (SAIC) have been very helpful during my time at Illinois. They opened the 'crown jewels' of their company so that I could have office space and the excellent tools associated with Xpatch. Dennis made sure that I always had priority support from his team. I am thankful for the data and tools provided for me by Dr. Steve Kosanovich of SAIC.

After the first year, the bulk of the coursework was complete. I had more time to devote to this research, but finding the right topic was not trivial. In December 2001, I took control of the ScaleME project. Dr. Sanjay Velamparambil helped me to learn and understand the fast multipole method and understand the code that

he authored. Without Sanjay's help, I would not have been able to accomplish so many neat things in my research. I am thankful for his support and encouragement throughout the research phase of my work.

Some of the essential research that helped me slice to the core of this technology was a project that Professor Chew gave me to study the error in the translation operator. He wanted me to extend some of the 2-D work of Dr. Shinichiro Ohnuki to 3-D. We met together and Shinichiro helped tutor me in his new approach to error control. It was a great project carved out and properly scoped to provide me with success. I learned a lot from the revision process involved in producing a journal paper. This was my first journal publication and a big milestone in my academic career.

I found out just before my preliminary exam that this paper had been accepted for publication. In preparing for my prelim exam, I had to write a document that discussed my research up to that point. Eric Dunn helped review my manuscript and provided excellent inputs, which improved the quality. In addition, Dr. Chew helped to review the work and enhance the readability. One big accomplishment leading up to the prelim exam was the solution of a 20-million-unknown sphere. This was made possible by the acquisition of the SUN Blade cluster in our Center. I thank Dr. Michielssen and Dr. Shutt-Aine for their efforts in acquiring and establishing this resource. The cluster was pivotal in my work.

As larger problems were solved, the noise error began to emerge as a prominent problem. Professor Chew hired a computer scientist, Howard Sun, who helped map out the large-scale scattering program so we could track the errors. Howard helped in compiling and in setting up a configuration control system for code revisions. I also received help from Lijun Jiang in programming, using the debugger, and other technical aspects. Lijun patiently and meticulously helped me many times. I also thank my office mate, Matt Delaquil, for his help in programming syntax and technical aspects. Matt has become a good friend and fellow researcher. We have shared and discussed many ideas together. For a short period, I shared the office with Dr. Eric Forgy. His conversations with me were technically stimulating.

In the final stretch of my research, I continued to receive support from the members of our laboratory. Andrew Hesford helped me overcome some LaTex problems. In working on the noise error bug, Dr. Jiming Song visited our Center and spent some time with me. His insights helped save me months of research

time. He knows more about FISC than anyone and I would consider him one of the leading experts in MLFMA technology. I should also recognize the help of Dr. Robert Chao and Dr. Jun-Sheng Zhao, who have left the Center. They both helped at various times in my research with Matlab codes and other questions I had. I thank Alaeddin Aydiner for his computer administration support. That is a tough job, as anyone who has been an administrator knows.

Returning to the contributions of the members of my committee, I thank Dr. Jin for helping me to understand the finite element method. He was also supportive of my work and had a good understanding of the Air Force constraint to finish within the allotted three-year time limit. I appreciate his thought-provoking questions that increased my understanding of electromagnetic theory. I appreciate Dr. Cangellaris for his willingness to be on my committee and his encouraging words during the process at times when I lacked the confidence in my ability to conduct the research.

I would not have been able to do this research without the support from my family and friends. Scott Graham and his family have helped me many times. Scott helped me to build an office in my home to study and his wife, Dashelle, watched our children during the final exam. My wife, has been my biggest supporter and I know things have been rough on her with our children and a workaholic husband. She, more than anyone, is looking forward to my graduation. Her contributions are enormous and these last three years would have been impossible without her help. She has planned the parties and food at each milestone and made sure that I was nourished regularly. She has made great sacrifices along the path. I love her and appreciate her. I also recognize the support and prayers of our immediate family members. I thank my parents and my wife's parents for their encouragement during this challenging assignment. They have done so much to get us where we are in life.

I am grateful for the department secretaries and especially for Karen Chitwood for her administrative support. I also thank the publications office for their editing work in the final production of this document. They have provided rapid, high-quality editing.

As a final note, I express gratitude to God for the comfort He has provided to me during these three challenging years. I believe that He placed the right opportunities in my path and the right people to help me along the way. He

helped me to stay healthy and motivated to diligently pursue this education and research. I believe that He prepared me at many stages of life to be able to earn a Ph.D. I hope that I may dedicate my life to helping other people and using this degree for the benefit of mankind. For those individuals not mentioned in this section, who have helped me, I hope that they receive heavenly recognition and know that I appreciate their help.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background

For many years, the gap between low frequency and high frequency methods in electromagnetics (EM) analysis has been wide. Both computing power and computational methods have improved. Computing power, including memory and processor speed, has been rapidly increasing and the fast multipole method (FMM) was born in the last decade and is revolutionizing scattering computations for surface scatterers. This rapid technology movement is enabling an overlap of low and high frequency methods. It is now possible to solve surface scattering problems with over 20 million unknowns. To put this in perspective, it is now possible to solve for the scattering of a full-scale fighter aircraft at X-band (8-12 GHz)! Solving such a problem is the state-of-the-art and is clearly not a routine process yet.

Using asymptotic techniques, the radar cross section (RCS) of such a target could be calculated to reasonable accuracy at many aspect angles down to L-band (1-2 GHz). Of course, many asymptotic codes do not account for second order effects such as multiple diffractions and generally return poor results for cavities and antennas. Since high frequency solutions have been the primary computational technique for acquiring needed data on electrically large complex structures, the question of accuracy is of paramount concern. Measured data have been used to determine the accuracy of such computational methods. Measuring targets at a range has been an imperfect art at best but has improved significantly over the last decade. The tongue in cheek quote [1], 'For confidence in a set of RCS data, measure once!' still applies. However, measurements are critical for developing truth data sets for benchmarking codes.

Measurements are done indoors and outdoors with static or dynamic targets. The targets can be physical models of actual targets and may even be smaller scale models. Static targets must be placed in the quiet zone. This is the region of nearly constant phase and amplitude where the incident wave illuminates the

target as if it were in the far field. Mounting techniques include low cross section pylons, foam columns, or string suspension. Depending on the target, frequency, and mounting method, the measured data could have significant contributions from the mount and its interaction with the target. Sometimes it is desirable to measure a target in a certain background. For example, a tank might be measured on the ground under some foliage. However, in general, it is useful to first understand the scattering mechanisms of a target by itself. Unlike static measurements, dynamic measurements are those done while a target is in motion. An airplane is a prime case where dynamic measurements are performed. Such measurements take into account the rotation of engine blades and the movement of control surfaces. As one can imagine, there are many possible measurement scenarios and target configurations.

Using different measurement techniques requires different methods of calibration and error control. Naturally, the dynamic measurements are the most difficult to control but usually have the highest correlation to reality for moving targets. Static measurements allow careful control of aspect angles and target configuration but suffer from error-inducing clutter such as mounting devices. As a radar heats up, its performance varies and its calibration is more difficult due to a change in amplitude and phase. Because of this natural effect, periodic calibration must be done. Primary calibrations before and after a measurement provide an idea of the system drift. However, periodic secondary calibrations should be done to ensure that the illuminating waves are constant over time. With all these imperfections and challenges in measuring targets, one might wonder if truth data can be established.

Rigorous theoretical formulation and application provide the key to fully understanding the scattering characteristics, called signatures, of a target. Using theory and good modeling practices, measured data can be understood and interpreted. Only through careful study comparing measured and theoretical signatures, can truth data be validated. The purpose of the forthcoming chapters is to carefully study the obstacles in applying the state-of-the-art theory to large-scale scattering problems. The method studied is called the multilevel fast multipole algorithm (MLFMA) and is second to none in calculation speed for a full-wave solver of a large surface scatterer. Using this algorithm, it is possible to better understand the scattering mechanisms of targets in frequency bands where asymptotic solutions

begin to lose validity and break down.

After covering the main parts of FMM (Chapter 2) and its extension to MLFMA (Chapter 3), the errors found in the application of the theory are considered. A way to better control one error source at the root of MLFMA is shown in Chapter 4. After examining the error sources and providing a framework to enhance error control, a discussion of several codes is given in Chapter 5. First, the well-known high frequency code called Xpatch is discussed. Then the premier MLFMA code, called the Fast Illinois Solver Code (FISC), is reviewed. Finally, the Large Scale Scattering Program (LSSP) used in this research is described. LSSP is a parallel MLFMA code based on the ScaleME (Scalable Multipole Engine) library. In Chapter 6, a description of a long-standing known bug in LSSP is given. The process that led to extermination of this bug is described in depth. This chapter highlights an educational component of debugging code. The scaling properties and efficiency of LSSP are studied in Chapter 7. Solving large-scale problems requires massive computer central processing unit (CPU) time and large computer memory. These requirements lead to the need for effective parallel implementation. In Chapter 8 a discussion of large-scale problems and optimization is given. Chapter 9 focuses on a innovative technique to reduce the memory through target rotation. Finally, Chapter 10 examines a missile-like target called the pencil to better understand the limitations of high frequency codes and the ability of full wave solvers to bridge the technology gap using MLFMA. Extensive data is included in Appendix A, and Appendix B contains some notes on the LSSP code.

1.2 Contributions

Systematic incremental contributions to science and technology by multiple sources govern the advancements in technology. Although most researchers strive for the quantum leaps, few are able to conduct research that makes previous technology obsolete. The more common result is that the research becomes another drop in a sea of a seemingly infinite knowledge base. This fact can be quite frustrating and demoralizing until one examines the bigger picture. Part of this big picture is that research facilitates the educational process. As scientists perform research, they increase their personal knowledge base and insights. This increase in understanding and knowledge augments their ability and potential to make future

contributions. Finally, incremental contributions often become the stepping-stones to the major advancements.

Using the contribution of a better error control method [2] in the heart of MLFMA, previous 2-D work was extended to 3-D problems. This extension is described in Chapter 4 and was published in *Microwave Optical Technology Letters* [3]. An initial study of the scalability of the parallel code, LSSP, was presented at the *2002 IEEE Antennas and Propagation Symposium* [4] and is described in Chapter 7. Further research was conducted to study different geometric targets and problems with more unknowns. An important discovery was the implementation of shared levels to dramatically increase the efficiency on multiple processors. Since memory and processor times are critical parameters in solving large-scale problems, ways to estimate these parameters accurately are very important.

The level of complexity of parallel MLFMA is reflected in the additional parameters required to set up a run. These additional parameters, such as the number of processors, have an impact on the memory and time required for a job. A new way to estimate the memory prior to a run based on the run parameters has been developed. Without the ability to understand and accurately estimate the memory usage, problems above 10 million unknowns are impossible. This memory estimator has enabled the solution of ultra-large-scale problems, like a 200λ sphere with 20 million unknowns shown in Chapter 8. That particular run doubled the previously established world record of 10 million unknowns, but first, problems with 12 and 15 million unknowns had to be conquered. Several obstacles were overcome in the pursuit of solutions of larger problems. The code to build a target sphere was limited to a maximum of 12 million and needed to be modified. Crossing the 15 million mark required attention to the column widths used for the facet count since they ran together when the eighth digit was added for 10 million facets. As far as geometry goes, crossing the 30 million mark will require more than 10 million nodes, which may present a similar problem. Of course, finding sufficient memory for a problem with this many unknowns will be a huge challenge that time will undoubtedly solve. Many of the obstacles found in solving increasingly larger problems are unique to large-scale problems.

Another significant contribution is the demonstration of our ability to solve large-scale problems on distributed memory machines. Using MLFMA with shared memory machines, like Silicon Graphics (SGI) machines, is simpler, but being able

to solve such problems on distributed memory machines significantly extends the utility of the technology. The 20 million sphere was solved with a new SUN Blade 2000, 10-node dual-processor cluster which has distributed memory. Figure 1.1 shows a picture of the cluster at the University of Illinois. It was acquired and set up during the fall of 2002. Additional distributed memory platforms that this technology could be demonstrated on are IBM SP, DEC Alpha, and low budget Linux clusters.

Figure 1.1 Ten-node CCEM SUN Blade 2000 cluster on rack.

Several problems arise when solving large-scale problems that do not have an impact in small-scale problems. Finding ways to address these problems is necessary to push the state-of-the-art forward. File sizes grow and are hard to manipulate. Runtimes and memory requirements are huge. Small errors in precision add up and can be seen in the output data. Determining the source of these errors and fixing them is an unquestionable contribution. Creative ways to fit problems into the memory were found and developed including target rotations to reduce memory.

Finally, the development of the pencil target and its use as a benchmark to study the limits of high frequency methods was useful. The data is valuable for proposing hybrid techniques to employ MLFMA at aspect angles where high frequency approximations break down. This set of data and analysis led to the observation of an anomaly in the horizontally polarized Xpatch bistatic data that will require further investigation.

The world of parallel MLFMA and its implementation are rich in discovery opportunity. As the rest of the computational electromagnetics (CEM) community tries to catch up with the University of Illinois (UI) in this area, our continued efforts to push the technology forward will maintain our cutting edge. The contributions described above are examples of why UI is leading in CEM.

1.3 Summary

As the state-of-the-art continues to march forward, further contributions will improve parallel MLFMA and increase the value of this technology to tackle real-world problems. It is important to study the performance of the code implementing parallel MLFMA so that it can be documented sufficiently, in order that the technology can be transitioned to a broad base of users. Finally, we must be vigilant in studying the error sources and discover clever ways to control the errors while managing the computational complexity.

CHAPTER 2

FAST MULTIPOLE METHOD

2.1 Introduction

This chapter is designed to give the reader a better understanding of some of the intricate details involved in the fast multipole method (FMM). FMM has been compared to a telephone switching network [5] in which hubs are used to transmit information rather than direct lines connecting each individual telephone. This greatly reduces the number of wires that must connect each telephone. The method can also be compared to a postal service process in which letters are passed between individual mailboxes. By creating a complex process including stations and special transportation carriers, multidestinational letters can be sent from one box to multiple boxes very efficiently compared to a brute force method of hand delivering to individual boxes. The process involves collection, sorting, bulk transportation, resorting, and distribution. The structure of the process allows for the efficiency. In FMM the structural process includes aggregation, translation, and disaggregation. When FMM is applied to a scattering problem the purpose is to reduce the computational cost of a matrix vector product. A scattering problem is formed with basis functions whose coefficients are unknowns. Each of these basis functions interacts with the others based on a known incident wave. If there are N basis functions, or unknowns (as we call their coefficients), an NxN matrix will describe the interactions between each basis function. The computational load of a matrix vector product is $O(N^2)$ and FMM allows this to be reduced to $O(N^{1.5})$. The critical step in FMM is the diagonalization of the translation matrix which Rokhlin [6] first realized. This translation step is similar to the mass transportation of grouped postal letters between post offices. This chapter will describe the terminology of FMM.

2.2 Basic Process

In the method of moments (MoM), an integral equation is discretized and can be written as a matrix-vector product. The matrix must be filled with the proper elements and then inverted to yield the solution vector. Direct inversion is an N^3 process, but by using a method like the conjugate gradient method it can be reduced to $O(N^2)$. Of course, for large-scale problems, this order is too high and must be reduced. The iterative approach of the conjugate gradient method in solving for the unknowns also requires many matrix-vector products. FMM is an enabling technology to reduce the computational workload of the matrix-vector product associated with these electromagnetic problems. As will be discussed in Chapter 3, MLFMA is the miracle algorithm that extends FMM and creates a truly enabling technology for large-scale EM problems.

Since the idea of MoM is to compute the interaction of each source with itself and every other source, FMM promises to streamline this process by allowing faster mathematical interactions. To understand how the basic building block of FMM differs from MoM, it is useful to visualize two collections of particles, each enclosed by a circle (in 2-D) as shown in Figure 2.1. In the spirit of MoM, each particle must interact with every other particle and itself. This is the well-known two-step process of filling and solving the impedance matrix. In the illustration, MoM would fill the matrix by interacting each point with itself and all the other points. This is done by starting with a point (like i_1) and computing the interaction with $i_1, \ldots, j_1, j_2, j_3, \ldots$, where the first three interactions with j-points in the right side box are illustrated with dotted arrows pointing to j_1, j_2, and j_3. The self-interaction terms fall along the diagonal of the matrix. In FMM, great savings are realized by eliminating the need to calculate all of these interactions directly.

The process to achieve massive computational savings requires three steps: aggregation, translation, and disaggregation. Let us call the left box of Figure 2.1, containing nine points, the source group and the right one containing ten points the field group. Aggregation involves the collection of all the source particles to the center of the source group, and this collection can be used to represent outgoing waves that appear to emanate from the center of the group. These waves are valid outside a circle enclosing all the aggregated sources. The smallest circle guaranteed to enclose all particles in a box is the one that intersects each of the box corners.

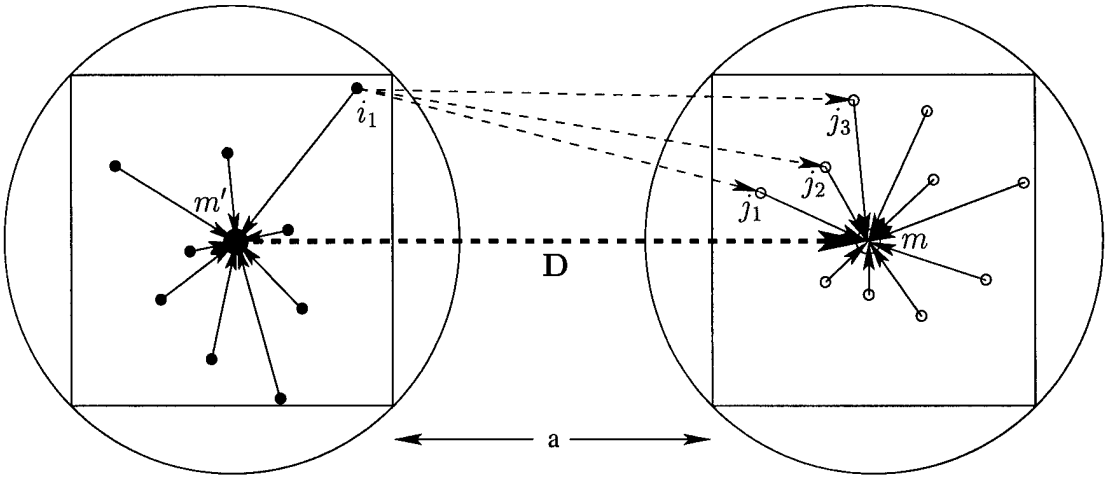

Figure 2.1 Two-dimensional source points and field points in aggregation, translations, and disaggregation. The square on the left contains the source points while the square on the right contains the field points. The three-step process includes the aggregation of nine sources from the left, translation of their multipole representation, and disaggregation to ten field points on the right. A single buffer square separates these two squares. The translation goes between the centers of the source groups and field groups and is shown with the vector **D**.

The outgoing waves are then translated to a field group center and distributed to each field point. Translation takes the radiation pattern of the sources and converts it to incoming waves valid inside a field group. This incoming pattern is used to represent the effect of each source on each field point. The distribution of the information to field points is the last step of disaggregation.

The addition theorem is fundamental in the upcoming mathematics. In order not to violate the addition theorem, the circle enclosing the source points cannot intersect the one around the field points. If it did, any particles found in the intersection region could not be considered due to this violation. For this reason, all boxes touching a source box including the source box itself must be calculated in the traditional way, using MoM. These neighboring boxes are sometimes called buffer boxes. In Figure 2.1, the buffer square of side length a is not shown but separates the same-size left and right boxes.

Near interactions done in the traditional MoM way can be reduced by decreasing the box sizes down to a minimum box size determined by accuracy constraints. The points, $i_1, \ldots, j_1, \ldots$, discussed above, actually represent the common edge centers of the basis functions. RWG (Rao, Wilton, and Glisson) rooftop triangular

basis functions are used to represent the geometry of the surface scatterers being solved. The center of the common edge between two adjoining triangles forming an RWG basis function determines into which box it belongs. The smallest box generally has a side length larger than $\lambda/4$ where λ is the wavelength, but depending on target discretization, it may be better to limit the size to 1.5 times the average edge length of the triangle basis functions. Average edge lengths are generally around 0.1λ, but edge lengths in general can vary between typical values of 0.02λ and 0.2λ. If the geometry discretization is too coarse, the surface currents cannot be accurately represented. If they are too fine, we run into low frequency breakdown problems along with excessive unknowns to solve [5].

In summary, a target is discretized with an appropriate basis function. These basis functions are sorted into groups. The basis functions in and around a group are calculated using MoM. All the other interactions are calculated using aggregation, translation, and disaggregation. Using this three-step process, the computational complexity is reduced to $O(N^{1.5})$ [5].

2.3 Aggregation

The grouping of sources in the first step of FMM is called aggregation. Aggregation is simply the summation of all sources into a radiation pattern emanating from the center of each group. This far field radiation pattern represents the sum effect of the sources contained in each box. This far field pattern is valid outside of the circle (2-D) or sphere (3-D) containing the source square (2-D) or cube (3-D). Of course, the number of samples required to effectively represent these far field patterns depends on the diameter of the circle or sphere enclosing the sources. More samples are required for larger boxes to capture a richer pattern. As will be discussed next, each unique radiation pattern is used to translate information to the appropriate field boxes.

2.4 Translation

In FMM for 3-D, there are at most $3^3 - 1$ cubes touching any given box. For a given source box, including itself, there are a maximum of 27 boxes (one buffer box case) requiring near interaction calculations. The rest of the boxes beyond the buffer boxes use the process of translation to account for the interactions.

Translation is the bridge over which outgoing waves are converted into incoming waves. Since this takes a far-field radiation pattern and changes it into incoming waves, it is often given the name 'outgoing to incoming' (O2I). It is actually a far-field to near-field transform function. This stage is based on the addition theorem and can be regarded as a dense or full matrix operation. However, through careful manipulation, this dense matrix can be diagonalized. The diagonalization of the translation operation is what gives the method the desired computational acceleration [5].

Fundamental to FMM is the addition theorem. Since real world large scale problems in EM generally require a 3-D approach, the addition theorem for the Green's function [5] in 3-D is given by

$$\frac{e^{ik|\mathbf{D}+\mathbf{d}|}}{|\mathbf{D}+\mathbf{d}|} = ik\sum_{l=0}^{\infty}(-1)^l(2l+1)j_l(kd)h_l^{(1)}(kD)P_l(\hat{\mathbf{d}}\cdot\hat{\mathbf{D}}). \tag{2.1}$$

In this equation $j_l(kd)$ and $h_l^{(1)}(kD)$ are spherical functions of the first kind (Bessel and Hankel, respectively) and $P_l(\hat{\mathbf{d}}\cdot\hat{\mathbf{D}})$ is the l-th order Legendre polynomial. Their arguments will be defined shortly but are based on the positions of the source and field points. Using the identity,

$$j_l(kd)P_l(\hat{\mathbf{d}}\cdot\hat{\mathbf{D}}) = \frac{1}{4\pi i^l}\int_0^{2\pi}\int_0^{\pi} e^{i\mathbf{k}\cdot\mathbf{d}}P_l(\cos(\alpha))\sin(\alpha)d\alpha d\beta \tag{2.2}$$

where $\cos(\alpha) = \hat{\mathbf{k}}\cdot\hat{\mathbf{D}}$ and $\hat{\mathbf{k}} = \cos(\beta)\sin(\alpha)\hat{\mathbf{x}} + \sin(\beta)\sin(\alpha)\hat{\mathbf{y}} + \cos(\alpha)\hat{\mathbf{z}}$, we can substitute Equation (2.2) into Equation (2.1) to get

$$\frac{e^{ik|\mathbf{D}+\mathbf{d}|}}{|\mathbf{D}+\mathbf{d}|} \approx \frac{ik}{4\pi}\int_0^{2\pi}\int_0^{\pi} e^{i\mathbf{k}\cdot\mathbf{d}}\sin(\alpha)\sum_{l=0}^{L}(-1)^l(2l+1)h_l^{(1)}(kD)P_l(\cos(\alpha))d\alpha d\beta \tag{2.3}$$

where the integration has been swapped with the summation, which has been truncated to L. With these equations in place, we can apply them to a source group and a field group. In applying Equation (2.3), it is appropriate to illustrate how a single source point is translated to a field point. In Figure 2.2, the vectors relating the source and field points to the previous equations are given as $\mathbf{r}_{ji} = \mathbf{r}_{m'i} + \mathbf{r}_{mm'} + \mathbf{r}_{jm}$, $\mathbf{D} = \mathbf{r}_{mm'}$, and $\mathbf{d} = \mathbf{r}_{m'i} + \mathbf{r}_{jm}$. Therefore, $\mathbf{D}+\mathbf{d} = \mathbf{r}_{ji}$ from Equation (2.3). This figure also shows the aggregation of a point i to the center m' and the translation from m' to m and then disaggregation from m to the field point j.

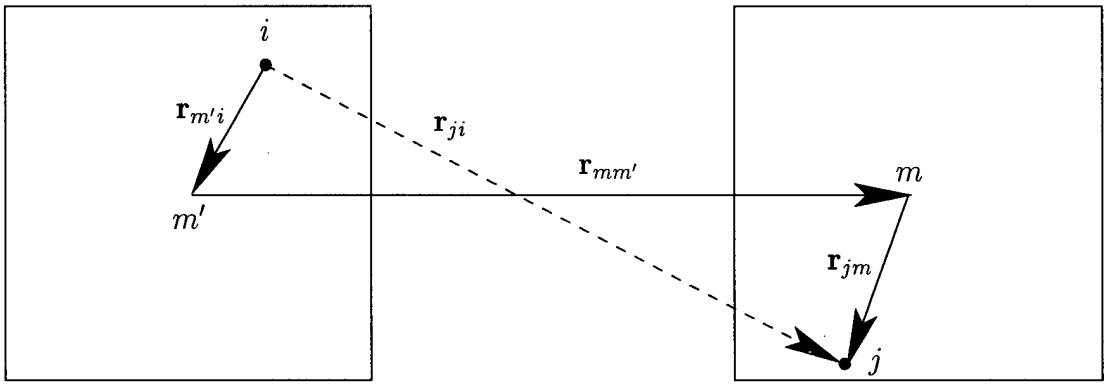

Figure 2.2 Source point and field point vectors in two boxes. The box on the left contains a source point i while the box on the right contains a field point j. Two paths from the source point to the field point are shown. The trivector path from i to j is used in FMM.

We can now give the scalar Green's function in terms of the translator, $T_{mm'}$, as

$$\frac{e^{ikr_{ji}}}{r_{ji}} = \int_0^{2\pi} \int_0^{\pi} \sin(\alpha) e^{i\mathbf{k}\cdot(\mathbf{r}_{jm}+\mathbf{r}_{m'i})} T_{mm'} d\alpha d\beta \tag{2.4}$$

where

$$T_{mm'} = \frac{ik}{4\pi} \sum_{l=0}^{L} i^l (2l+1) h_l^{(1)}(kr_{mm'}) P_l(\hat{\mathbf{k}} \cdot \hat{\mathbf{r}}_{mm'}). \tag{2.5}$$

The integration is performed over the unit sphere and α corresponds to θ while β is the integration around ϕ. The integration over the unit sphere is performed using at least $2L^2$ Gaussian quadrature points. In implementing the integration, two processes are used to simplify the integral. First, the translation vector $\mathbf{D} = \mathbf{r}_{mm'}$ is aligned with the $\hat{\mathbf{z}}$-axis such that $\hat{\mathbf{k}} \cdot \hat{\mathbf{r}}_{mm'} = \cos(\theta)$. Secondly, the $\sin(\alpha)$ is removed by transforming the integration over $d\alpha = d\theta$ by letting $u = \cos(\alpha)$ and $du = -\sin(\alpha)d\alpha$. Equation (2.4) then becomes

$$\frac{e^{ikr_{ji}}}{r_{ji}} = \int_0^{2\pi} \int_{-1}^{1} e^{i\mathbf{k}\cdot(\mathbf{r}_{jm}+\mathbf{r}_{m'i})} T_{mm'} du d\beta \tag{2.6}$$

and Equation (2.5) becomes

$$T_{mm'} = \frac{ik}{4\pi} \sum_{l=0}^{L} i^l (2l+1) h_l^{(1)}(kr_{mm'}) P_l(u) \tag{2.7}$$

The integration over the new variable u involves the multiplication of an L-th order Legendre polynomial with an exponential function. The order of this part of the integrand can be considered to be approximately $2L$ because it is bandlimited, and Gaussian quadrature allows for exact integration of a signal of order $2L$ using $L+1$ Gaussian points. The integration over $\beta = \phi$ is over a complete period and requires $2L$ points. Thus, in total, $2L^2 + 2L$ points are required for the integration. It is important to note that the translator $T_{mm'}$ is independent of the source and field point vectors. This independence allows for a diagonalized translation operator. The translated radiation patterns are then used to interact with the individual field points.

2.5 Disaggregation

The disaggregation process is usually done at the same time as the translation operation. In this way, the effect of the radiation patterns of grouped sources is applied to individual field points. Every box, in turn, becomes a field box receiving the information from distant source boxes through translation. For a more comprehensive treatment of FMM technology see [5]. In the end, each point corresponding to a basis function can be found and the resulting coefficients used to calculate important target characteristics such as RCS. In FMM, the computational complexity is $O(N^{1.5})$ [5]. To improve the computational complexity, it is necessary to take the algorithm to a new level.

CHAPTER 3

MULTILEVEL FAST MULTIPOLE ALGORITHM

3.1 Introduction

This chapter extends FMM described in the previous chapter to a multilevel algorithm named the multilevel fast multipole algorithm (MLFMA). An integral part of the extension to a multilevel scheme requires both interpolation and anterpolation of the radiation patterns between differently sized boxes. The size of the box determines the levels. Tree structure, levels, upward pass, downward pass, and filtering (including interpolation and anterpolation) will be discussed. The advantage of extending FMM for the Helmholtz equation to multiple levels is the reduction in computational complexity from $O(N^{1.5})$ to $O(N \log N)$ [5], where N represents the number of unknowns describing the target.

3.2 Tree Structure

To reduce the computational cost, the number of translations used in FMM must be reduced. This requires a tree structure connecting boxes of different sizes. In the case of FMM, the translations are between equally sized boxes. Therefore, the finer the boxes, the more translations must be made. With MLFMA the number of translations can be reduced by aggregating radiation patterns from smaller boxes to form larger box radiation patterns that are translated to larger-size boxes with appropriate separation. This is easiest to see using a 2-D grid as shown in Figure 3.1 where two example source squares, shown as shaded, are translated using MLFMA. In Figure 3.1, the grids have been produced by successive subdivisions. Level 1 includes the four squares produced by the first subdivision. Translations cannot be done on level 1 because all boxes touch each other. Level 2 contains 16 squares and it is the highest level where translations can be performed. In MLFMA, we want to perform translations at the highest level possible. Translation is only possible to squares that are not touching. Level 3 has 64 squares and in this example it will be the lowest level.

In Figure 3.1, we assume that every square contains information that must be related to every other square in the most efficient manner. On the left column, a source square in the lower left part of the grid is shown with 7 translations between level 3 squares. Rather than performing 48 more translations on this level, the pattern from this source box is combined with the neighboring three patterns as shown in the middle diagrams to form a larger square box pattern which only needs to be translated 12 times to the centers of larger level 2. On the right column of Figure 3.1, a source square is translated to 27 level 3 squares. None of these 27 squares qualifies for higher level 2 translations. Using a single buffer square, 27 is the maximum number of translations for a given level from a single square and is $6^2 - 3^2$. However, there are $7^2 - 3^2$ possible translation vectors on a single level from arbitrary source squares. The remaining 28 translations can be performed on level 2 using just 7 translations. Furthermore, these 7 translations include the information aggregated from three other level 3 source squares. Computational savings, through a more efficient translation process, are an important feature of the multilevel algorithm. The reason that more levels are desirable is because lower level translations are more efficient than near interactions computed using MoM.

The center of a box is important and the algorithm revolves around these centers. In a way, the centers can be considered command centers. This analogy fits particularly well with the multilevel aspect of the algorithm. A command center has a commander who is responsible for subordinates. Commanders are also required to communicate directly with colleagues on an equal level and superiors on a higher level. After describing how the command structure is created, the details in a command context will be addressed.

When MLFMA is implemented, the program must first create a tree structure that resembles an organization chart. In 2-D, a quad-tree is created by successive binary cuts in both directions. In 3-D, an oct-tree is created in which a cube around the target is subdivided into eight equal cubes. The number of cubes at any level is given by $2^{\mathbf{N_{dim}}*\mathbf{L_{lev}}}$ where $\mathbf{N_{dim}}$ is the dimension (2 or 3) of the problem and $\mathbf{L_{lev}}$ is the level number. In Figure 2.1 there are $2^{2*3} = 64$ boxes on level 3 of the 2-D space. Since the target must be contained inside the square at level 0, the maximum dimension A of the geometry in Cartesian space is used as the level 0 length. Thus, the length of the lowest level is given by $A/(2^{L_{low}})$, where L_{low} is the number of the lowest level. As described in Section 2.2, this has a minimum

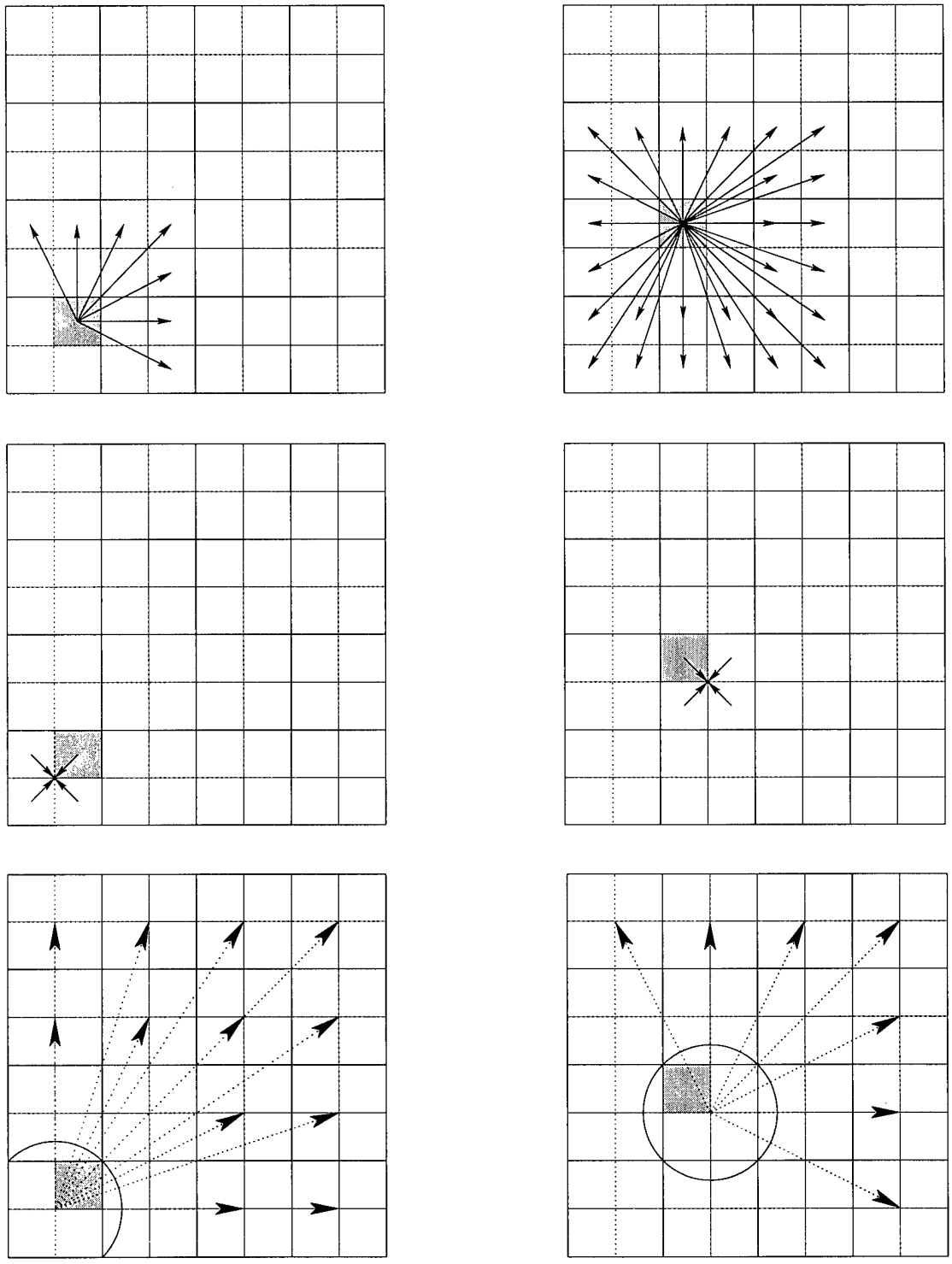

Figure 3.1 Two examples of multilevel translations. The top shows translations on level 3. The middle figures illustrate aggregation of radiation patterns to larger level 2 boxes. The bottom shows level 2 translations.

which should carefully be considered with regards to the average size of the basis functions. Therefore, a maximum level exists. Pushing the tree structure to the maximum level maximizes the computational efficiency of the algorithm.

Since the target bounding box is known, the number of levels can be chosen and the size of the smallest boxes can be found. The basis functions, indexed by spatial points and loosely referred to as particles, are then sorted into these smallest boxes. Naturally, many of the boxes are empty. The tree is pruned of these empty boxes. Each box will have an average number of particles based on the total number of particles divided by the number of filled boxes. The number of particles in any given box will vary. With the assumption that a target's surface is flat as it cuts through a box and that the angle of the cut is a uniform random variable, it is possible to calculate that 1.9 out of 4 squares in the 2-D case will be crossed on the average. In 3-D when a cube is subdivided into 8 cubes at a lower level, empirically we observe that about 4 out of the 8 cubes are filled with particles. Obviously, this can be very geometry dependent and if the number of particles in a cube gets close to 4 or less, pushing additional levels will not continue this trend. In the limit, if every box contains only a single particle, subdividing the box will produce only one single filled box.

With multiple levels and a pruned tree of filled boxes, the command structure and organization is in place. In the context of a command structure, it is possible for a commander to have $2^{N_{dim}}$ filled boxes, but generally only half of these boxes are filled. The next sections will discuss the steps taken on each box.

3.3 Traversing the Tree

Each filled box must be considered in order to capture the total interaction of each particle with every other particle. Recall that the purpose of MLFMA is to accelerate the matrix vector product without requiring the full matrix storage. Rather than causing individual particles to interact with each other, we now have a tree structure in place to handle the interactions. Traversing the tree begins at the lowest level.

Each lowest level command section or box has some particles. The first step is to compute the direct interaction of these particles with themselves and each other. Then each of the neighboring boxes and their particles must be considered.

After accounting for all of these near interactions, the particles in each box are aggregated to the center of the box, which we will refer to as the commander (CC). The commander is then responsible to create an outgoing radiation pattern from the aggregated sources. The CC then must determine which commanders should receive this information through translation. These are the boxes outside of the buffer boxes, which cannot receive higher-level translations.

The upward pass begins by shifting the radiation pattern up one level in the chain of command. This communication requires special upsampling so that the commander's supervisor can integrate the radiation patterns from the subordinate boxes and then create an appropriate radiation pattern for their level. Since the higher level command box has twice the edge length of the subordinate commander boxes, higher-level CCs need more samples to describe a richer radiation pattern. On levels that are above the lowest level, or smallest boxes, near interactions are not issues. The key is to collect the information from the subordinate command boxes and then to translate to appropriate same-level commanders separated in the usual sense. For efficiency, each CC will shift their patterns to their supervisor who will continue the same process. This process of translating and shifting goes until level 2 is reached and the effect of the original sources at the lowest level of command has been provided to the entire space.

In this process, it is easy to see that upper level commanders have more details to work with as they collect the information of subordinates and pass it across the tree or up the tree in the most efficient way. Although the upward pass ends at level 2, the process is not complete. We must now consider the downward pass and its objective to provide the needed information to every lowest level commander for distribution to each of the field particles. In the downward pass, translated information is received and the outgoing radiation patterns are converted to incoming wave patterns and combined with other incoming wave patterns. These incoming wave patterns are then filtered or anterpolated down to the subordinate commanders who combine information from the top levels with translated information from same level or colleague commanders. The resulting patterns are formed and the process of passing the information down the tree continues until the lowest level commanders have the required data to interact with each of their field points. This completes the process of traversing the tree.

In summary, each lowest level CC has facilitated the interaction of each of their

source particles with the field particles of every other lowest level commander. This was accomplished through direct interactions for self and near neighbors, aggregation, translation, upshifting, downshifting, and disaggregation. The algorithm is not difficult to understand, although the mathematics and finer details tend to cloud the big picture. As in command, however, the finer details always remain important.

3.4 Signal Processing

Probably the most important finer detail of MLFMA is signal processing. This is also essential in the principle of command. In the context of MLFMA, signal processing includes the upsampling (interpolation) and downsampling (anterpolation) of radiation patterns. Filtering is necessary because the radiation patterns of finer lower-level boxes do not require as much detail or samples as coarser higher-level boxes. Therefore, when a CC provides a radiation pattern up the chain of command, it must first be interpolated and then shifted to the next level CC. The interpolation allows the higher level CC to receive the pattern with the correct sampling points.

On the downward pass the process of anterpolation allows the lower level CC to receive a downshifted pattern with fewer but an adequate number of sample points. These are easily combined with translated outgoing to incoming patterns from colleague commanders with the same number of samples. This continues down to the lowest level commanders that distribute or disaggregate the incoming wave pattern to each field point.

3.5 Final Details

This chapter has described MLFMA. The detailed mathematics including the incorporation of MoM have been intentionally left out. A good treatment of the subject can be found in the first three chapters of [5] where the derivations for both the electric and magnetic field integral equations are given in detail. Also, the mathematical details of shifting, interpolation, and anterpolation are provided. With this background, it is now possible to understand the error sources of MLFMA. As expected, such a fast method will come with some tradeoffs between speed and accuracy.

CHAPTER 4

ERROR CONTROL

4.1 Introduction

When using MLFMA to produce RCS data, there are several key error sources. Depending on the type of targets and the accuracy required, these errors can be controlled to some degree. Understanding and controlling these errors is the purpose of this chapter. Considering a perfect electric conductor (PEC) target, these errors fall into the following classes: geometry modeling, integral equation discretization, matrix equation solving, and translation operation. The translation operator and its factorization are at the core of the multilevel algorithm and together form an important part to understand. This chapter will be primarily focused on the effect of this translation error and a new approach to error control. The recently developed new approach for 2-D problems [2] has been extended in this research to 3-D problems [7] and will be the main focus of this chapter.

4.2 Geometry Modeling

There are differences between a computer model and the physical model that it represents. Targets with flat surfaces are the easiest to model and usually have the most faithful representation. For example, a cube can be modeled perfectly with just eight points. However, all physical cubes will have rounded edges and corners due to the tolerances of the physical manufacturing process. Depending on the frequency of interest, these differences may be quite insignificant. For curved surface models, a linear representation is an approximation and will therefore be an error source. Since most large-scale problems involve complex structures, there will be a difference between the geometrical representation and the actual target.

These complex targets may be formed by the combination of flat surfaces joined with curved and doubly curved surfaces. When basis functions are chosen based on flat triangular patches, the surface can be meshed carefully to capture the key scattering structures. Of course, narrow tips result in triangles that are too

small or narrow for good computational stability. This means that the geometry modeler must have a clear understanding of the effect of the small details and how to represent them.

Many companies interested in EM modeling hire Ph.D.'s to work on CAD modeling. They apply EM theory in the construction of models that are most suitable to a particular EM code simulation. Constructing the geometry in an optimal way that captures the physics of the problem requires a solid understanding of computational electromagnetics. For curved surface models, higher order basis functions that describe the curvature can be used to more faithfully represent the shape, but the tradeoff is increased complexity. The point is that geometry fidelity and choice of basis functions lead to an important error source. This error source is one of the hardest to quantify.

4.3 Integral Equation Discretization

Another source of error arises from MoM. Since MLFMA is tied to MoM, this particular error source should be mentioned. Integral equation discretization is the process of converting an integral equation into a matrix equation. The way this is generally done is by using a basis function as defined over the geometry surface. Then a testing function is used and integration over the basis functions and testing functions is performed to find the matrix elements in MoM. The error associated with integral equation discretization or converting a continuous integral to a discrete matrix is not studied in more detail in this document.

4.4 Matrix Equation Solving

A discretized integral equation can be represented as a linear system of equations

$$\bar{\mathbf{A}} \cdot \mathbf{x} = \mathbf{b} \tag{4.1}$$

in which the matrix $\bar{\mathbf{A}}$ is the impedance matrix, $\mathbf{x}$ represents the unknown coefficients of the chosen basis functions, and $\mathbf{b}$ is a known vector based on the incident wave. For small MoM problems, the matrix can be inverted and then multiplied by $\mathbf{b}$ on the right hand side of the equation to find $\mathbf{x}$. However, for large problems, the cost of matrix inversion is too high. These systems are solved with iterative

methods such as the conjugate gradient method. In such an iterative method, a solution vector is guessed and the matrix-vector product is compared to the known vector. Using the comparison, a new solution vector is chosen and the process is iterated until the comparison, as measured by what is called the residual error, becomes small enough. Often this residual error is chosen to be 10^{-3}.

This residual error means that the unknown coefficients are only approximate values. The unknown coefficients of the basis functions represent the surface currents. These surface currents are used to calculate the scattered field. The relationship between the error in the scattered field as a function of residual error is very dependent on geometry and aspect angles of both the transmitter and receiver. Fortunately, approximate currents often lead to reasonably accurate RCS values as the integration of the currents for the far field is a smoothing process. Another advantage of MLFMA is that it is matrix free.

4.5 Translation

The hardest error to control finds its root in the heart of FMM. The key to FMM is the diagonalized translation operation. As described in Chapter 2, the translation operator is repeated below in Equation (4.2). The derivation of the translation operator used in 3-D MLFMA is given in many sources [5, 8, 9, 10, 11, 12, 13] and is based on the addition theorem. The series representation of the translation operator must be carefully truncated to avoid excessive error. With insufficient terms, the series is a poor approximation, but with too many terms the divergent nature of the series emerges. Since computational methods require truncation for evaluation and due to the finite precision of floating point numbers, this truncation number must be carefully chosen. The series representation of the translation operator is given as [5]

$$T_{mm'} = \frac{ik}{4\pi} \sum_{l=0}^{L} i^l (2l+1) h_l^{(1)}(kr_{mm'}) P_l(\hat{\mathbf{r}}_{mm'} \cdot \hat{\mathbf{k}}) \tag{4.2}$$

where $h_l^{(1)}(kr_{mm'})$ is a spherical Hankel function of the first kind, $P_l(\hat{\mathbf{r}}_{mm'} \cdot \hat{\mathbf{k}})$ is a Legendre polynomial of order l, $\mathbf{r}_{mm'}$ is the translation vector, and $\hat{\mathbf{k}}$ is a unit vector used for integration over the unit sphere. This is the key series used to approximate the Green's function by integrating over the unit sphere using approximately $2L^2 + 2L$ Gaussian quadrature points and the relationship

$$\frac{e^{ikr_{ji}}}{r_{ji}} = \int d^2\hat{k} e^{i\mathbf{k}\cdot(\mathbf{r}_{jm}+\mathbf{r}_{m'i})} T_{mm'}. \tag{4.3}$$

Figure 4.1 is useful for visualizing the translation of a source point to a field point. This is an orthogonal view of two cubes and an outline of the sphere (diameter=$\sqrt{3}a$) enclosing them. The upper cube with labeled points has been rotated to capture the 3-D perspective. Note that this third cube could fit between the lower two cubes and could be a buffer cube. The **r**-vectors represent the vectors between group (or box) centers, field, and source points. The trailing subscript is the starting point of the vector. Thus, swapping subscripts is the same as negating the vector. Rather than calculating the direct path from the source to the field point $\mathbf{r}_{ji}$, shown as a dashed line, the source point is aggregated to the box center on the left, then translated to the box center on the right, and finally disaggregated to the field point. Mathematically, this path is $\mathbf{r}_{jm}+\mathbf{r}_{mm'}+\mathbf{r}_{m'i}$ and is represented with the solid vectors. The translation does not depend on the positions of the source and field points; however, the truncation of this series in Equation (4.2) is highly dependent on these positions. In this illustration, the source and field points are at opposite corners of their box groups. This produces the highest error in the evaluation of the scalar Green's function given in Equation (4.3). We refer to this worst case as position 81 where the digits represent the corner of the cube where the field and source points are located. Similar worst-case positions include {18, 27, 36, 45, 54, 63, 72}.

It is important to note that a sphere enclosing a source point cannot intersect a sphere enclosing a field point. This would violate the requirements of the addition theorem [14]. Thus, the adjacent boxes, including diagonal boxes, containing field points require a direct evaluation of the interaction. Therefore, the nearest translation is between boxes separated orthogonally by one buffer box and is twice the distance of the box edge length a, as seen in Figure 4.1. The error associated with the Green's function and its approximate representation is due to the position vectors of the source and field points relative to their box centers. Obviously, when their relative positions are the same, the approximation is the best, but when the vectors are maximum and in opposite directions, this leads to the largest error. As illustrated in Figure 4.1, $|\mathbf{r}_{m'i}+\mathbf{r}_{jm}|$ is at a maximum. In 3-D, this maximum length is $\sqrt{3}a$ and there are eight positions of source and field points where this occurs. This maximum distance can be used to predict the truncation number L,

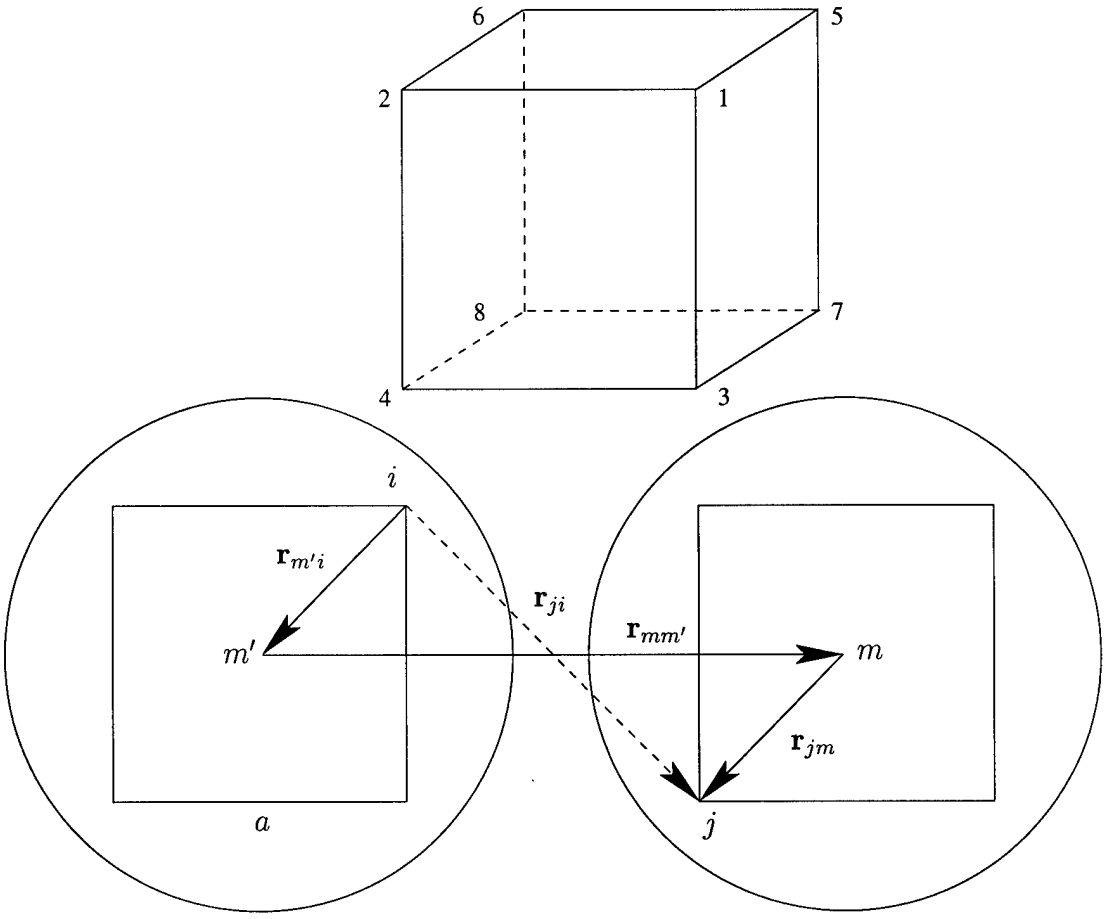

Figure 4.1 An orthogonal view of two cubes and the outlines of the spheres containing them. The top cube with corners labeled from 1-8 gives a perspective view. Source point i is at point 1 and field point j is at point 8. This is referred to as position 81.

in the series of the translator formula in Equation (4.2).

4.5.1 Series truncation

Previous work [5, 13, 15] applied the refined excess bandwidth formula,

$$L \approx kd + 1.8(d_0)^{\frac{2}{3}}(kd)^{\frac{1}{3}}, \tag{4.4}$$

in order to truncate the series for good results. In this formula, $d = |\mathbf{r}_{m'i} + \mathbf{r}_{jm}|$, the wavenumber is k, and d_0 is the desired digits of accuracy. For the worst case, $d = \sqrt{3}a$. It says that, for a given box size, a reasonable truncation L, can be calculated and used to terminate the series.

Figure 4.2 shows the actual error compared to that predicted by the excess bandwidth formula for increasing L. This example, for a fixed $ka = 20$, shows increasing disagreement for $d_0 > 2$ (relative error less than 10^{-2}). We also note that the minimum error occurs when the machine precision accuracy of $d_0 \approx 15$ is used in Equation (4.4). This is a key observation in developing the new approach. The excess bandwidth formula applies when $L < kr_{mm'}$ or, in other words, when the group centers are sufficiently separated relative to the truncation. By increasing the buffer boxes separating two groups, larger truncation values can be chosen. In this plot, one can easily see how initially after a certain number of terms ($L > \sqrt{3}ka > 34$ in this case), the error begins to decrease and after reaching an actual minimum (at $L = 67$), the divergent nature is seen. This also shows that the excess bandwidth formula can predict and achieve certain error levels; however, the desired number of digits of accuracy used in Equation (4.4) is not always achievable and is different from reality when the box separation and box size are small and high accuracy is desired.

Using the excess bandwidth formula in Equation (4.4) and the worst case position vectors in 3-D pointing to field and source points in opposite corners of the box (see Figure 4.1), we can solve for the digits of accuracy as

$$d_0 = \left[\frac{L - \sqrt{3}ka}{2.2(ka)^{\frac{1}{3}}} \right]^{1.5}. \tag{4.5}$$

This equation relates to the convergence of the Bessel function when it is $O(10^{-d_0})$. In the same way that d_0 represents the digits of accuracy with respect to the convergence rate of the Bessel function, using the asymptotic approximation of the spherical Hankel function [16] when $kr_{mm'} \sim O(l)$, it can be shown [5] that the divergence of the Hankel function series when it is $O(10^{+d_1})$ leads to

$$d_1 = \left[\frac{L - kr_{mm'}}{1.8(kr_{mm'})^{\frac{1}{3}}} \right]^{1.5}, \tag{4.6}$$

where d_1 represents the number of digits lost by numerical evaluation. This can be written in terms of the number of buffer boxes n and box size a as

$$d_1 = \left[\frac{L - (n+1)ka}{1.8((n+1)ka)^{\frac{1}{3}}} \right]^{1.5}. \tag{4.7}$$

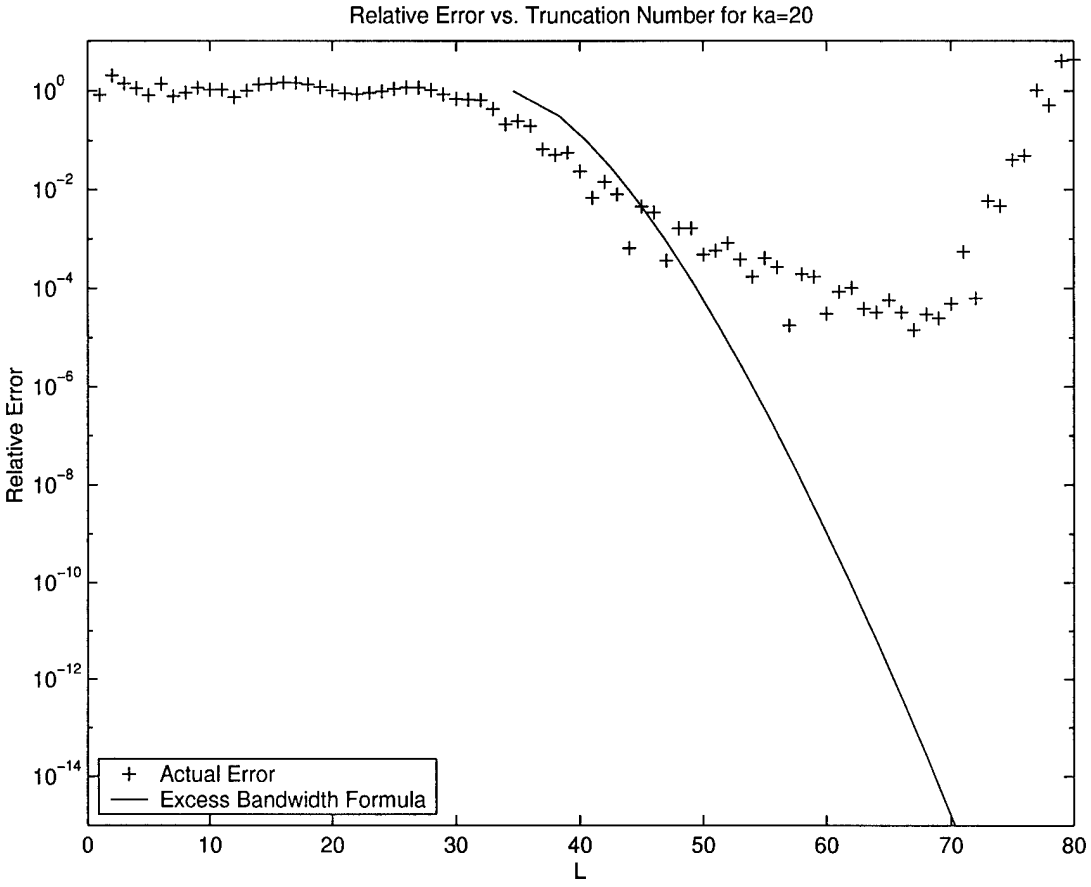

Figure 4.2 Error comparison for fixed $ka = 20$ and changing L. The optimal truncation is $L = 67$ for this one buffer box worst case. When $L > 67$, the series is clearly diverging. We note that theoretically $L > 40$, in this example, is outside of the valid range for applying the excess bandwidth formula although numerically it achieves a better error up to $L = 45$. Assuming that L is chosen using Equation (4.4) with 15 digits accuracy, the actual error does not even achieve 5 digits of accuracy. This is expected since we are also outside of the applicable range of this formula.

By subtracting $d_0 - d_1$, the actual expected digits of accuracy can be found. Since machine precision limits the digits of accuracy to about 15, for small box sizes, the digits lost will be larger when L is increased in Equation (4.7). Also, if the number of buffer boxes n is increased, then d_1 will shrink. Equation (4.4) produces L_{max} based on $d_0 = 15$ as

$$L_{max} = \sqrt{3}ka + 13.2(ka)^{\frac{1}{3}}. \tag{4.8}$$

L_{max} is the truncation number that should produce the minimum possible error. Since $d_0 = 15$ implies a relative error of 10^{-15}, the example in Figure 4.2 shows a large gap between the true minimum error and that predicted by the excess bandwidth formula. If $L > L_{max}$, the error grows due to the finite machine precision and the divergent series embedded in Equation (4.3). Using $L = L_{max}$ in Equation (4.7), we can find $d_{min}^{(n)} = d_0 - d_1$, which is the true digits of accuracy associated with the minimum error for n buffer boxes,

$$d_{min}^{(n)} = 15 - \left[\frac{L_{max} - (n+1)ka}{1.8((n+1)ka)^{\frac{1}{3}}}\right]^{1.5}. \tag{4.9}$$

To capture the effect of lost digits using one buffer box, $n = 1$, leads to a $d_{min}^{(1)}$ that will represent the theoretical error bound,

$$d_{min}^{(1)} = 15 - \left[\frac{L_{max} - 2ka}{2.3(ka)^{\frac{1}{3}}}\right]^{1.5}. \tag{4.10}$$

Obviously, it is best to choose $L \leq L_{max}$ since this represents the error floor. When $(n+1)ka < L < L_{max}$, the new approach is required for precise error control. It is desirable to calculate a better truncation number in this part of the controllable region, and this is what the new approach does. When $L < (n+1)ka$, the new approach uses Equation (4.4) to choose the truncation.

In order to choose the truncation properly in the extended controllable region where the excess bandwidth formula does not achieve the desired error level, the new approach is necessary. In the new approach, we find the truncation number based on a desired error level e_r and the intersection of this error level with the minimum error level and the border where the excess bandwidth formula applies (see [2, 17, 18] and Figure 4.3).

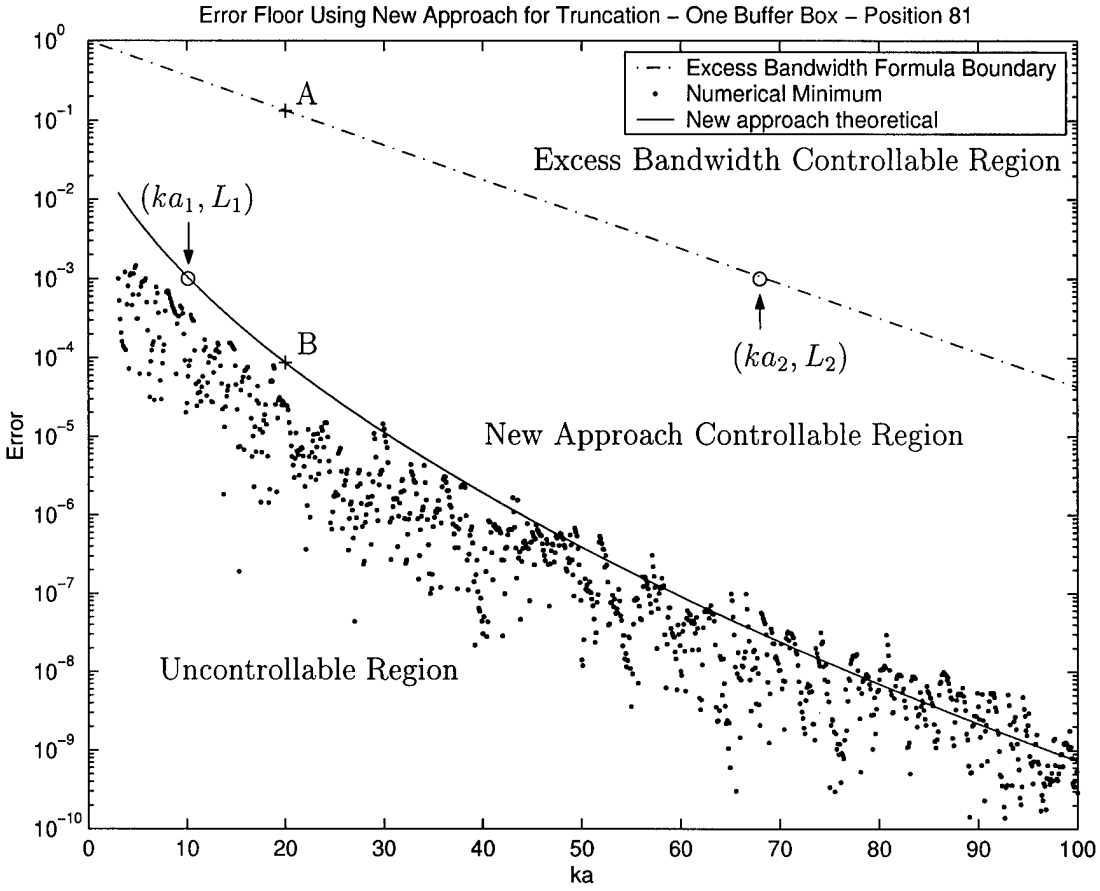

Figure 4.3 Error floor using new approach for truncation and one buffer box. The boundary between the uncontrollable and controllable regions is shown as the new approach theoretical minimum line. The upper dash-dot line is the boundary between the upper region where the error can be controlled using the excess bandwidth formula and the middle region where the new approach must be used for precise error control. For a fixed ka, for example $ka = 20$, the error level can be controlled to levels between point A and point B using the new approach. For an error level above point A, the excess bandwidth formula can control the error. Finally, for this $ka = 20$, we cannot theoretically control the error to levels below point B.

From these two intersection points, (ka, L) pairs are found. For both points, the same desired error level is used to find (ka_1, L_1) and (ka_2, L_2) as illustrated in Figure 4.3 for $e_r = 10^{-3}$. The first ordered pair, on the left, borders the uncontrollable region and is given by

$$ka_1 = \left\lceil \frac{13.2 - 1.8(n+1)^{\frac{1}{3}}\left(15 - \log\frac{1}{e_r}\right)^{\frac{2}{3}}}{n+1-\sqrt{3}} \right\rceil^{1.5} \tag{4.11}$$

and

$$L_1 = \sqrt{3}ka_1 + 13.2(ka_1)^{\frac{1}{3}}. \tag{4.12}$$

These equations are derived by solving Equation (4.9) for ka_1 where the subscript refers to the first ordered pair. Of course, L_{max} from Equation (4.8) must be substituted into Equation (4.9). Above, L_1 is simply L_{max} from Equation (4.8) with ka_1. The second ordered pair is given by

$$ka_2 = \frac{3.2\log\frac{1}{e_r}}{\left(n+1-\sqrt{3}\right)^{1.5}} \tag{4.13}$$

and

$$L_2 = (n+1)ka_2 \tag{4.14}$$

and meets the boundary where the excess bandwidth formula can be used. Equation (4.13) comes from setting $(n+1)ka_2$ equal to the right hand side of Equation (4.4) with $d = \sqrt{3}a_2$. Then the equation is solved for ka_2. Here, L_2 forms the upper boundary of the new approach controllable region. Interpolating between these two points provides an adjusted L necessary to maintain the required relative error level for a given ka between ka_1 and ka_2. Using this new approach, the error can be controlled for 3-D translation operations in FMM and MLFMA.

4.5.2 Results

The new approach allows us to establish the minimum possible error for a given box size. If the desired error is higher than this minimum, the error can be controlled using the new approach. If the desired error is found in the region where $L < (n+1)ka$, the refined excess bandwidth formula given in Equation (4.4)

is used to determine the proper truncation number. When $L > (n+1)ka$, the new approach must be used to control or minimize the error. All of the numerical results discussed here involve the worst case errors in which the source and field points are at opposite corners of their boxes and with only one buffer-box group separation.

In Figure 4.3, we show that the minimum error for position 81 (worst case) can be found by setting $d_0 = 15$ in Equation (4.4). This gives the L_{max} of Equation (4.8) that can be substituted into Equation (4.10) to find the theoretical minimum error. This is labeled in this figure as the 'new approach theoretical.' Numerical results are given where the numerical minimum error is found compared to the theoretical predicted minimum error. Figure 4.3 shows that the numerical minimum results tend to be slightly lower than the theoretical minimum for smaller ka; this is due to the approximate form of the refined excess bandwidth formula. The variation of the true minimum is due to rounding. Since the true minimum was desired, the lowest error was found through numerical experimentation. It matches the new approach theoretical line fairly well. Previously, it has been shown [2] that the excess bandwidth formula applies when $L < (n+1)ka$ where n is the number of buffer-box separation between the groups. This upper-bound line is the boundary between the two controllable regions.

As a function of the distances between box centers, the minimum error for the worst case can be predicted for boxes just outside the buffer box layer. Note that the maximum distance ($\sqrt{12}a$) for the first layer beyond the one buffer box is greater than the two buffer box minimum distance ($3a$). The translations on a level will have lower errors when the ratio of the translation distance to the box size increases. This is shown in Figure 4.4 where the legend represents the number of boxes in each orthogonal direction ($\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$) to get to the field box from the source box. There are $5^3 - 3^3 = 98$ different translation directions one buffer box away. However, Figure 4.4 shows the relative error associated with these six unique distances.

Using interpolated values of L we can establish the predicted error level inside the extended controllable region where the excess bandwidth formula breaks down. Figure 4.5 shows constant error levels where the excess bandwidth formula is used, and in the region where interpolated values of L are used, the theoretical error level dips downward. This is because the interpolation picks L to be slightly more

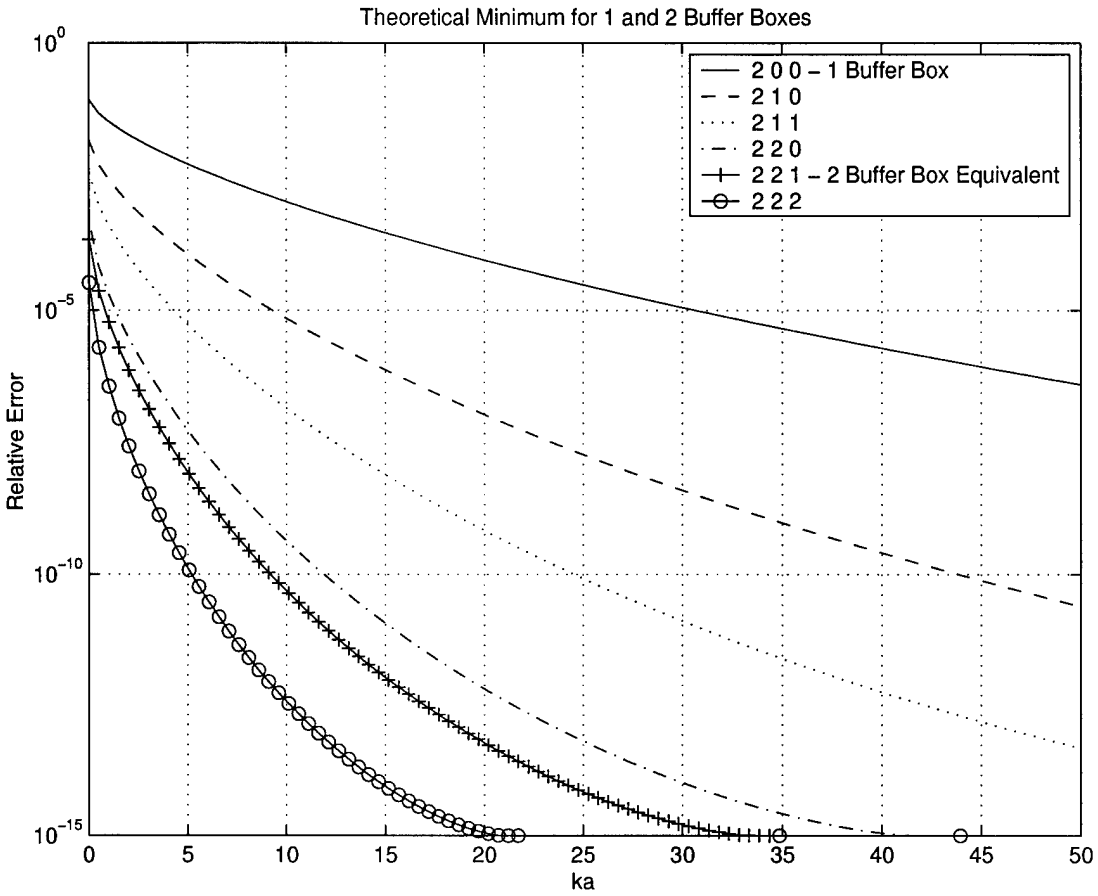

Figure 4.4 Predicted minimum error for different translation distances. There are 98 different translation directions one buffer-box layer away. However, as this plot shows, there are only six unique distances. For example, (2 2 1) corresponds to (**x**, **y**, **z**) and is a distance of three times the side length of the box.

than necessary to achieve the fixed error level.

To illustrate the new approach, Figure 4.6 compares the new approach to the excess bandwidth formula with an arbitrary three digits of desired accuracy. It is easy to see the two boundaries between the three regions of chosen L values for varying ka. When $ka > 67$, we use the excess bandwidth formula. When $ka < 10$, we cannot make the error smaller than the desired error level, so we use L_{max} to achieve the best possible error. In the middle region we use interpolated values of L as described in the formulation. This plot highlights the utility of the new approach in achieving a better error control than the excess bandwidth formula.

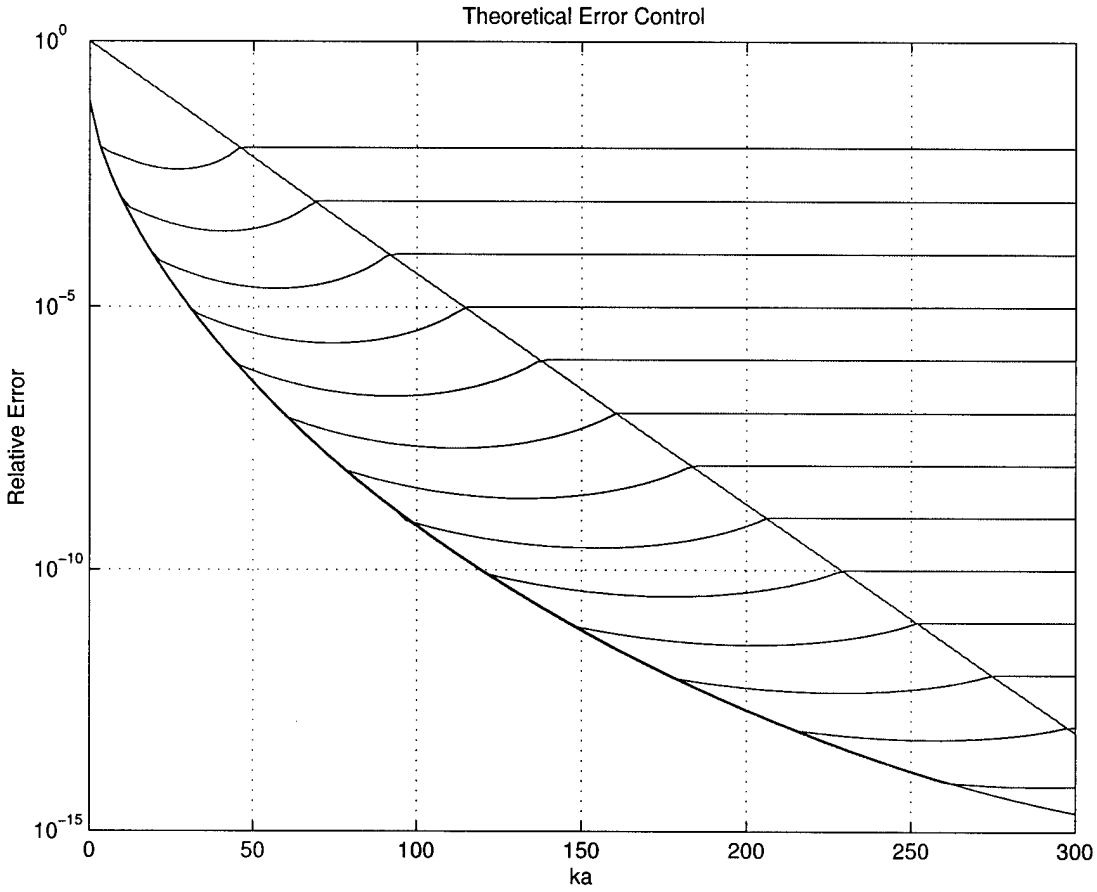

Figure 4.5 Theoretical error levels using carefully chosen truncation values. The downward curvature indicates too many terms used in series. Digits of accuracy range from integer values of 2 to 14.

The truncation numbers corresponding to the theoretical minimum error in Figure 4.3 are shown in Figure 4.7 as the upper solid $L = L_{max}$ line. The boundary line between the upper two controllable regions represents the $L = 2ka$ line. This corresponds to the lower solid line in Figure 4.7. From Figure 4.7, it is clear that the lower error achieved by the new approach requires more terms in the series. This figure also has both the values of L chosen by the new approach and the excess bandwidth formula for $d_0 = 3$ (the example illustrated in Figure 4.6). Naturally, the values of L chosen when $L < 2ka$ are the same for the new approach and the excess bandwidth formula. This is seen at $ka > 67$ where the lines intersect and become one. Interpolation is used when $2ka < L < L_{max}$, and L_{max} is used in the

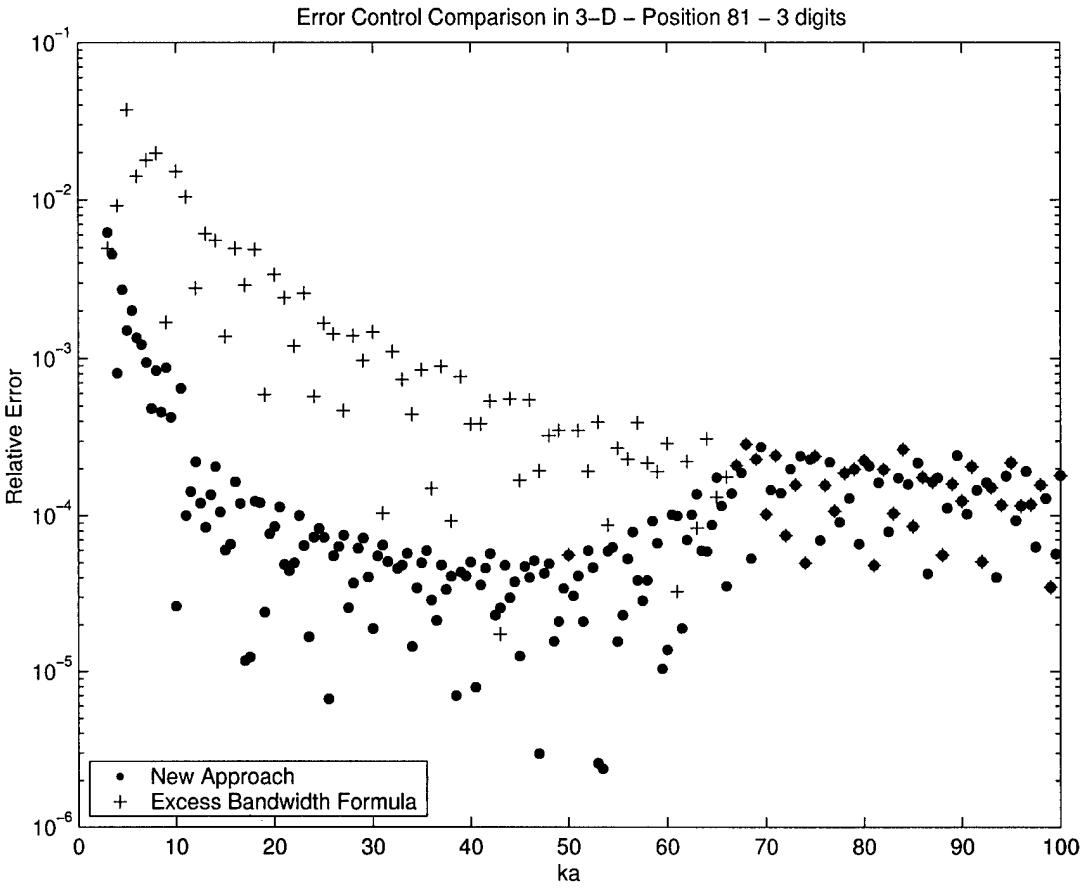

Figure 4.6 Comparison between the new approach and the excess bandwidth formula for a desired three digits of accuracy. The new approach uses the excess bandwidth formula for $ka > 67$ and the minimum error for $ka < 10$. The difference highlights the need for the new approach in the region $(n+1)ka < L < L_{max}$.

uncontrollable region to minimize the error.

Figure 4.8 compares the truncation number when $3 < ka < 10$ for the actual minimum error which is around 10^{-3} and the L selected using numerical results. The deviation is $+1$ to -2 between the predicted L that would achieve the minimum error and the actual L values that are used for the minimum error. In other words, the minimum was often found using slightly fewer terms than predicted. This figure also shows the number L found by the excess bandwidth formula with $d_0 = 3$ to show how many more terms are required to achieve the minimum error as compared to what the excess bandwidth formula predicts.

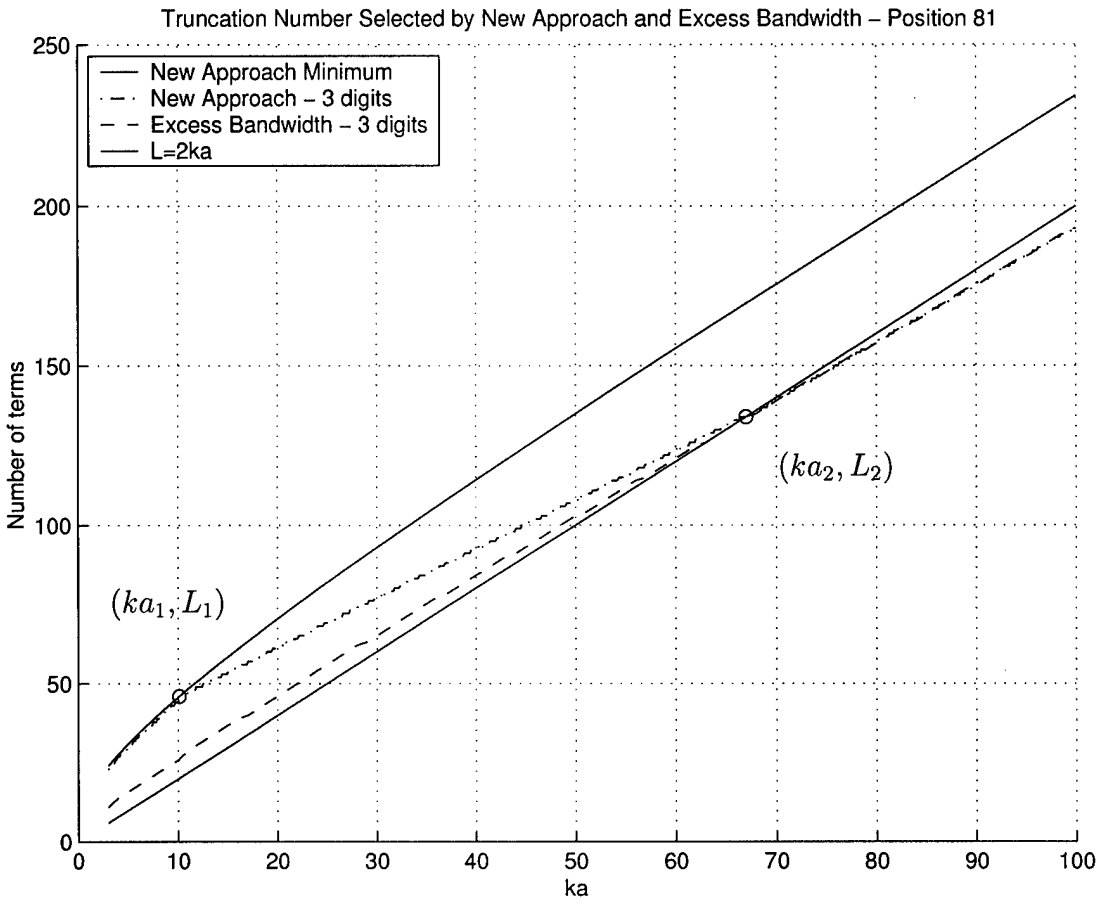

Figure 4.7 Truncation number for varying ka. The theoretical number of terms needed by the new approach to achieve the minimum error for varying ka is compared to the number of terms selected using the excess bandwidth formula with $d_0 = 3$. The line for one buffer box where $L = 2ka$ is shown, and it intersects the line where the new approach is used to control the error ($d_0 = 3$).

Finally, the new approach and excess bandwidth formula are used together to achieve a fixed error for increasing ka. In the extended error controllable region, the interpolated values of L are used to simulate the relative error using the new approach. Figure 4.9 shows the numerical data for fixed error levels as ka is changed. In each case, the required accuracy is achieved. Using higher order interpolation in the region where the new approach is used would lead to a better prediction of L and, hence, to a flatter response. To use higher order interpolation would require root finding of the midpoint truncation L_{mid} and curve fitting between (ka_1, L_1),

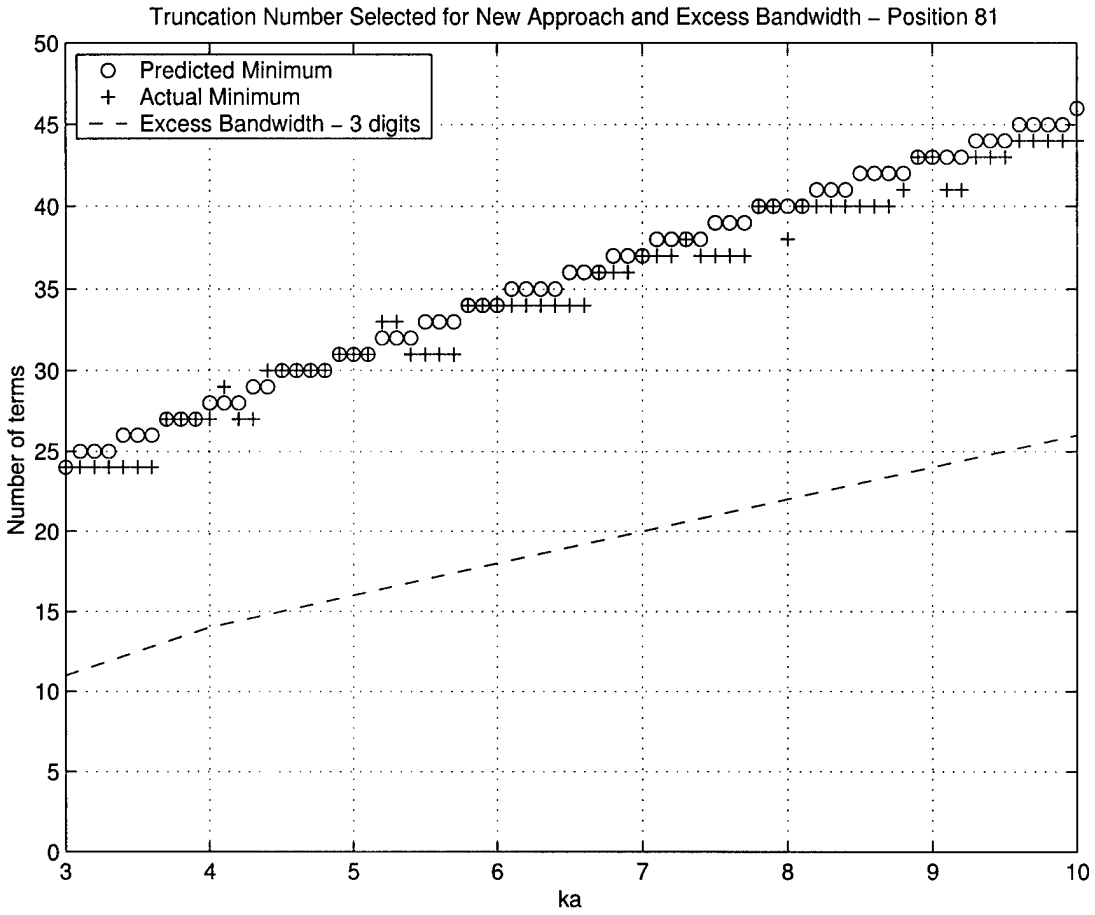

Figure 4.8 Actual versus the predicted number of terms to achieve the minimum error. Only small differences are seen. The number of terms selected using the excess bandwidth formula with $d_0 = 3$ is also shown for comparison.

(ka_{mid}, L_{mid}), and (ka_2, L_2).

The point of these results is to show that the error can be better controlled to achieve the desired accuracy using the new approach as a companion to the excess bandwidth formula. Of course, accuracy in the region where the error is uncontrollable can only be improved by increasing the buffer boxes which is a tradeoff to efficient calculations.

4.5.3 Better error control

When the truncation number is chosen using horizontal interpolation, where the interpolation is done using (ka_1, L_1) and (ka_2, L_2), we see that the theoretical

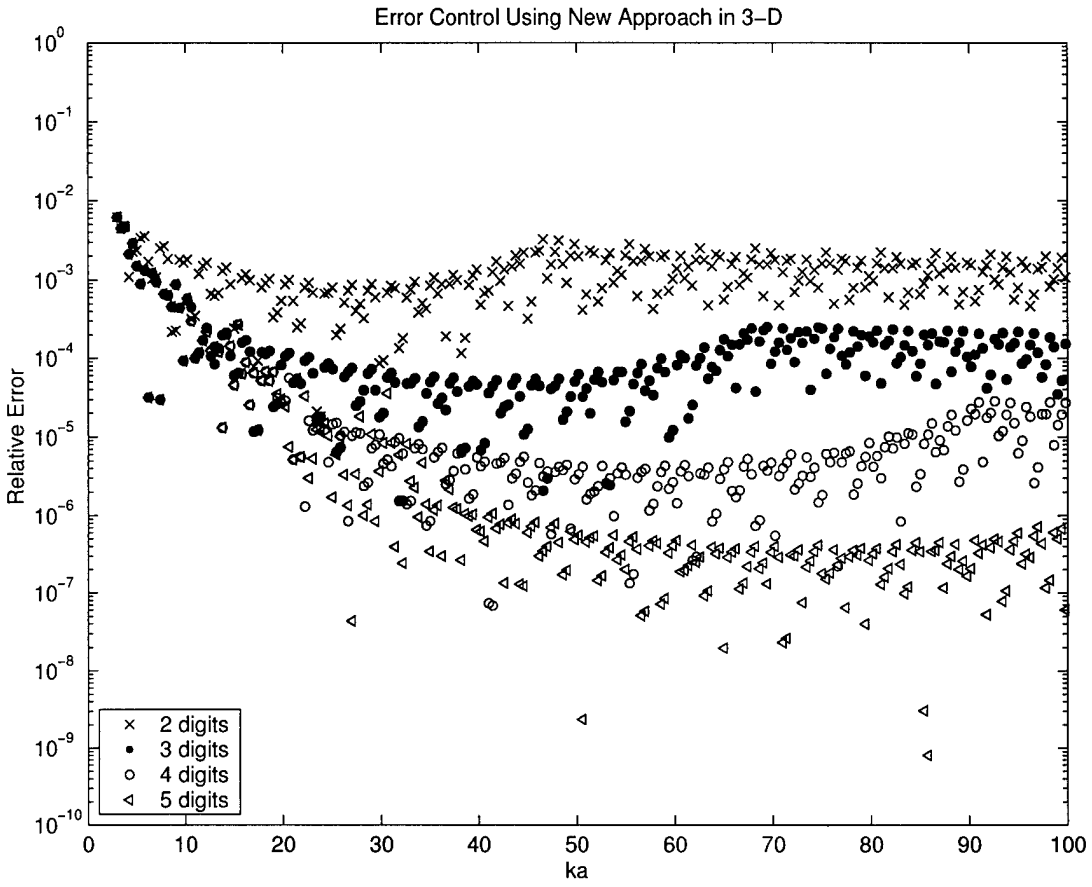

Figure 4.9 Numerical results using the new approach to control error level. Fixed error levels are achieved by employing the new approach. The legend gives the number of digits accuracy desired. The slight dip in the extended error controllable region where the new approach is used can be eliminated through higher order interpolation [2].

error is lower than predicted using linear interpolation. Of course, the error will even be lower when the source and/or field points are closer to their box centers or near each other's relative position. The bending down of the error is due to larger truncation, L. Using a root solver, a precise truncation can be found to place the error at a fixed point. However, since the horizontal span of the minimum error line and the error line for $L = 2ka$ at a fixed error level is larger than the vertical span, vertical interpolation should be better. Figure 4.10 shows the theoretical error for fixed error levels using vertical and the previously discussed horizontal

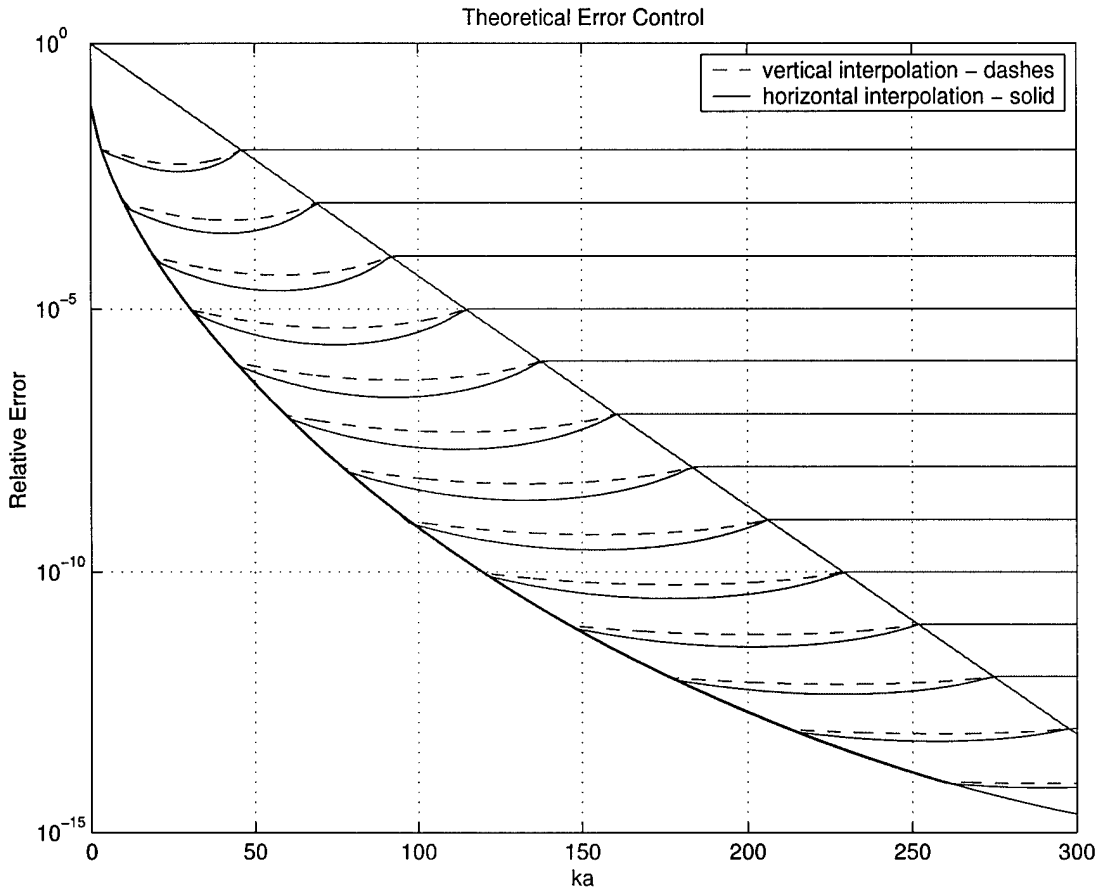

Figure 4.10 Vertical interpolation versus horizontal interpolation. The vertical interpolation produces a flatter response but still has the typical dip.

interpolation. Referring to Figure 4.3 the vertical interpolation simply uses points A and B at a fixed ka but different error levels to interpolate the desired truncation number.

Looking at one last case as purely an academic exercise, we examine the collinear case where the source points and field points are actually outside the box but inside the sphere enclosing the box. We see that the collinear case is actually the worst case. Figure 4.11 shows the collinear case where the vector to the source point and to the field point are along the translation direction. Fortunately, we find that for practical problems, the error will always be lower than this collinear case.

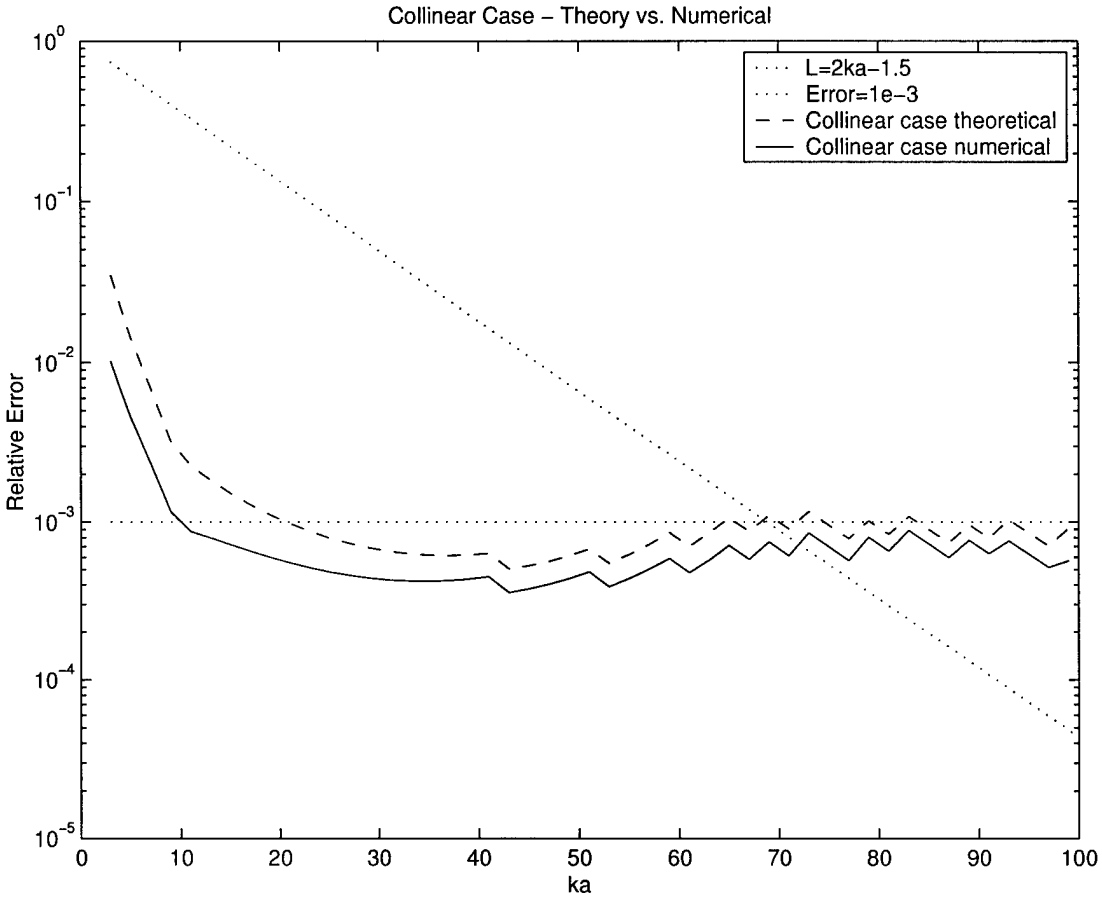

Figure 4.11 Vertical interpolation for the collinear case. The collinear case does not occur because the source and field points would fall outside of the boxes even though they are inside the group spheres.

This chapter has described some of the error sources of MLFMA and deeply explored the errors associated with the translation operator. The new approach can be used to predict and control the error.

CHAPTER 5

XPATCH, FISC, AND LSSP

5.1 Introduction

Xpatch, FISC, and LSSP are computer codes used for RCS calculations. They were created under the direction of University of Illinois faculty over the past decade and a half. Xpatch has roots back to 1988 when its creator, Dr. Shung-Wu 'Andy' Lee, developed the shooting and bouncing ray code designed to apply the physical theory of diffraction to calculate high frequency RCS scattering from complex objects like aircraft. The U.S. Air Force began sponsoring the development of the Xpatch code in the early 1990s in support of the Non-Cooperative Target Identification program (NCTI). At the time, NCTI was the number one critical technology listed by Air Combat Command. The development was funded through a start-up company called DEMACO, Inc., which was founded in 1986 by Dr. Lee and his wife. In 1998, Science Applications International Corporation (SAIC) acquired DEMACO and continues to operate the DEMACO division in Champaign, IL. A teaming effort to research and apply computational electromagnetics is on-going between the University of Illinois, Center for Computational Electromagnetics and SAIC.

The Fast Illinois Solver Code (FISC), based on MLFMA, was created under the direction of Founder Professor Weng Cho Chew in 1997 in collaboration with DEMACO, Inc. Although its development has not been funded close to the level that Xpatch has been funded, it provides tremendous capabilities and can be considered an industrial strength code employing state-of-the-art technology. Based on MoM, it is capable of very precise solutions. Using MLFMA allows the code to achieve a computational complexity of $O(N \log N)$. This allows FISC to solve millions of unknowns, thereby conquering problems previously unsolvable except through approximate high frequency techniques. FISC and its associated technology have successfully bridged the gap between traditional MoM codes and high frequency codes.

FISC requires shared memory and is most suited for Silicon Graphics (SGI) or SUN machines. It has been successfully parallelized on the SGI architecture. The Large Scale Scattering Program (LSSP) was created by Dr. Sanjay Velamparambil under the direction of Professor Chew and is the first successful attempt to create a truly parallel version of MLFMA independent from any particular MoM code. LSSP uses the MoM code called Trimom and the MLFMA library of ScaleME (Scalable Multipole Engine). It is designed to be highly portable to many architectures. In fact, it has been demonstrated on a networked heterogeneous cluster.

5.2 Code Usage

These three codes have been used and tested to different levels. Xpatch is the most mature and LSSP is in its infancy. Each of the codes comes with a wide variety of parameter settings that affect the output speed and accuracy. The cost of high accuracy can be quite high and it is common to expect the last 2 dB of accuracy to represent 80% of the cost. The precise relationship of the parameters to the accuracy is often target dependent. This is an area that could use more research. However, as simulated data is compared to measured data or analytical solutions when they exist, better understanding can be gained regarding accuracy and scattering mechanisms. Once the parameters and code limitations are understood, good data can be obtained.

Part of this research is designed to study the limitations, strengths, and weaknesses of these codes and the methods behind them. Most of the focus is on the latest code, LSSP, and studying its scaling properties as a parallel implementation of MLFMA. In the process, it is useful to compare LSSP with FISC as they are based upon the same technology. Since the MLFMA technology is capable of bridging the gap between high frequency methods and low frequency methods, we now have the ability to understand the actual limitations of high frequency methods applied to different targets at various frequencies and aspect angles. With precise MoM-based solutions, the validity of high frequency methods can be tested at lower frequencies where physical optics type scattering is not the dominant effect. As an example, to make it clear what this means, a missile target might be modeled at 10 GHz where its length is on the order of 100 wavelengths; however, at 3 GHz the solution may begin to break down. The solution might still be fine on

the broadside, but around nose-on incidence, the secondary effects might become the dominant scattering mechanisms.

5.3 Limitations

All codes have an operating range and parameters that make them valid within that range. People tend to push the limitations of the operating range with little understanding of the impact. The main impact is degraded accuracy. The problem is that usually the impact is hard to quantify. In Chapter 4, some of the error sources were described as well as the results of the translation error research.

When using integral equation based methods to produce data, we generally expect high accuracy. However, depending on the modeling and parameters chosen, the accuracy level will vary. If high accuracy settings are made, one expects longer run times but higher accuracy. Since the main purpose of this thesis is to study large-scale problems, we should have the ability to compare the output to simulations using high frequency methods. One of the limitations of the code LSSP is its restriction to PEC surface structures. It does not handle materials of any kind at this time. This is a severe limitation in the application to real world problems. Nevertheless, there is still plenty of work to be done to research PEC targets.

It is hoped that benchmark comparisons can be established between the three codes to demonstrate some of the unknown limitations of high frequency methods. The aspect of speed versus accuracy can also be addressed in such benchmark studies. Most importantly, recommendations for future hybridization can be made after studying these codes.

CHAPTER 6

LSSP DEBUGGING

6.1 Introduction

All computer codes, including those described in Chapter 5, have bugs; perfect codes do not exist. At the beginning of this project, the Large Scale Scattering Program (LSSP) was no exception – it needed the services of an 'exterminator' as much as any new code. The Fast Illinois Solver Code (FISC), a sister program to LSSP, utilized TRIMOM, an early MoM (method of moments) code, as its foundation. The developers of LSSP designed the ScaleME accelerator so that it would be compatible with any MoM code. Because TRIMOM could be easily incorporated into the context, and because it was developed at the Center for Computational Electromagnetics at the University of Illinois (CCEM) it was the logical choice.

As researchers at the University of Illinois pushed the state-of-the-art in large-scale problems, they found a disturbing difference between FISC and LSSP. This disparity emerged in the study of large-scale spheres with millions of unknowns and progressively worsened. In these scenarios, LSSP produced noisy data with fluctuations through areas of the bistatic scattering that were supposed to be very flat. This instability increased as target size and number of unknowns increased. Despite the efforts of several great scientists, the anomaly was elusive. LSSP had great potential to produce precise data quickly, so these researchers focused a high level of energy on eliminating this error. Where their learned efforts did not isolate the bug, my own diligence was rewarded with success. This chapter will detail my long process of discovering the source of this problem, the 'root bug' and other anomalies that I found along the way. In so doing, I hope to show fellow scholars that dedication can be the difference between victory and defeat. Finally, this chapter is written in a manner that highlights the educational component of debugging code. The debugging process that I endured helped me slice to the core of the technology, including the finest details.

6.2 Preliminary Background

My involvement with the LSSP code began on 13 December 2001 with this message from my advisor: "...it will be good for you to learn as much about ScaleME as possible before Sanjay leaves. Sanjay is planning on leaving in February. He just told me today." At the time, I was approaching the halfway point of the time that the Air Force had allotted for my PhD studies at the University of Illinois. I lacked only one class, which was scheduled for the following semester. I had almost no experience with the computational side of computational electromagnetics (CEM), nor did I have any degree of mastery over C and FORTRAN programming. I understood the basics of computer languages; however, the fruits of programming can only be picked by those who know the syntax. Fortunately, I had a strong background in UNIX systems and an even stronger desire to push the limits of the state-of-the-art through productive research.

Upon reading this memo, I contacted the code author, Dr. Sanjay Velamparambil, who gave me several papers to read and set up a time to talk about the code. He quickly ascertained that my grasp of the fast multipole method (FMM) was even more uncertain than my ability to interpret code syntax. My task in taking over the ScaleME project, as it is commonly called, seemed to be daunting and formidable, if not hopeless, when one considers the intellectual rigor and technical acumen of my predecessors. Sanjay told me that he was going back to India to get married, but he promised that he would come back in February and that I could always e-mail or call him for support.

The few days that we had together before his departure went by very quickly and I soon found myself alone, in charge of the ScaleME project. At the end of January 2002, I decided to record my work in a lab book, which has been a useful reference for several reasons. In the beginning of the project, this resource helped me clarify my direction. While I worked long hours at the Center, it showed my daily progress through the massive amount of code. Now, this record documents my revision process and allows me to take a glimpse of that mountain, which I climbed over and blazed a trail through in the course of this research work. On 11 February 2002, Sanjay returned as a married man. I was happy to spend some time with him and to get clarification and help on the project.

I devoted this introductory phase of my research to learning FMM and the

basics of running the code, which required me to first compile its nearly 50,000 lines. I also had to acquire several different accounts on supercomputers including the locally affiliated National Center for Supercomputing Applications (NCSA). These early struggles competed with my final academic class but I was able to make good progress. By the beginning of March, I was running and testing an IBM version of LSSP on a Department of Defense supercomputer at the Naval Oceanographic Office's major shared resource center, NAVO. The SGI Origin 2000 machines at NCSA and a local network of SUN workstations were also made available to me. I intended to push the number of unknowns beyond 10 million, but problems inherent in the program frustrated my efforts many times in the process.

LSSP proved to be very difficult to compile and to operate, crashing often and lacking the robustness of FISC. Until I began to realize that the code was a work in progress, my expectations could not be met. For example, when any operator requested a certain angle, a bug in the program caused it to use a different calculated angle, which in turn produced an incorrect output angle. However disappointing these errors were, this program, the product of four long years of development, had great potential as a research code. A parallel implementation of the multilevel fast multipole algorithm (MLFMA), ScaleME promised to be portable to low cost Linux clusters as well as heterogeneous clusters with different operating systems. Although even today it still lacks the ability to process materials other than perfect electric conductors, this flexibility is a leap forward in a technology still dominated by the old paradigm of huge supercomputers at limited locations. By moving to the interconnected node paradigm that has shown potential in the private sector, ScaleME increases the ability of smaller institutions, corporations, and even individuals to contribute to the state-of-the-art. With this vision in mind, my advisor recruited a talented computer scientist, Howard Sun, to help debug ScaleME. He mapped the subroutines into a complex flowchart that seemed to go on and on. It would be no easy task to track down inconsistencies and errors in this sea of code.

One day in August 2002, I came into my office and wrote on top of a piece of paper, 'Kill the angle problem.' By modifying four files and many of their subroutines and functions over the next several hours, I put this problem to rest. The program would now use the input file angles to calculate the scattering and then output those same angles with the data. The reason this problem was difficult to debug was that the variables for aspect and integration angles were almost indis-

tinguishable, having very similar names, like ntheta, nTheta, NTheta, nThetaG, etc. At a few key points in the code, some of these variables were being used interchangeably, with frustrating results. Howard had only missed a few details in fixing this bug and if he had been able to better differentiate between the variables, he would have surely identified the problem areas and achieved consistent results. This dilemma, very avoidable by the initial code developers, became the cause for systematic error in later versions. With the weight of such negative evidence to show what could go wrong, all code authors should take special care to choose more transparent variable names and track them systematically, in order to minimize confusion for future developers.

An equally crucial advance in the development of LSSP was Howard's implementation of a configuration version control system. This setup allowed me to keep closer tabs on the changes that I had made to this complex and dense code. I still came across instabilities and had problems on some of the supercomputers using large numbers of processors, but I was making progress in studying the scaling properties of LSSP. My expertise in both programming and the technology of MLFMA was rapidly maturing. By early fall of 2002, I began to see what this technology was all about. With a working knowledge of the code and a lot better understanding of MLFMA, achieved through studying the error associated with truncation shown in Chapter 4, I was ready to go deeper into the code.

6.3 Focusing on the Noise

After I passed my preliminary exams in January 2003, I met with Dr. Chew, my advisor and a Founder Professor. At that meeting, he first made it clear to me that finding out why ScaleME data was filled with noise compared to FISC could be the turning point to the success and proliferation of this technology. It was not the first time we had discussed this problem, but when he associated the term, 'significant contribution' with eradicating this particular error, I knew that we had found the right focus to complete my Ph.D. I did not then know whether it was possible for me to find the root cause of this noise error but within four months of our conversation, I had done it. Before I found and eliminated the problem, ScaleME generated noise of increasingly higher amplitude in direct correlation with the increase in numbers of unknown variables. Figure 6.1 shows

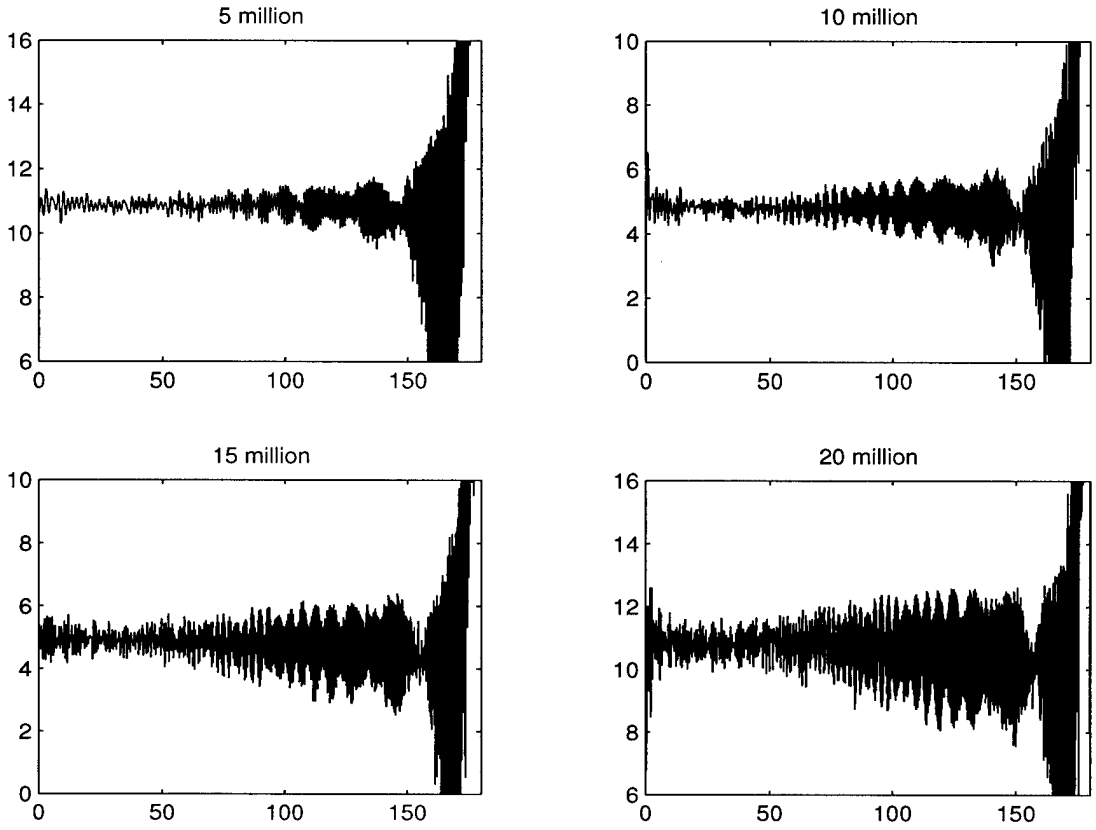

Figure 6.1 Large-scale sphere error growth.

the data for 5, 10, 15, and 20 million unknown spheres before the bug was fixed. This figure illustrates how the amplitude of the noise increases as the problem size increases in terms of number of unknowns.

Although I did not know if I could discover the elusive bug behind this error, I was determined to give it my best effort. I had an increasing sense of purpose and clarity as my search for the noise error took me on a journey of great learning and personal growth. It solidified my understanding of MLFMA and the sea of details accompanying this technology. It also helped me understand, to a greater extent, the method of moments applied to three-dimensional problems with RWG basis functions.

6.4 Bug Isolation

When you search for the proverbial 'needle in the haystack,' you must begin somewhere based on: collected data, most probable problem areas, or simply a hunch. The first place we suspected an error was in the interpolation and anterpolation. This made sense as a good starting point because the data indicated that larger-scale problems suffered more noise error. These larger problems involved more levels, which would require more interpolation traversing up the tree and anterpolation on the downward pass. If an error could be found in these parts of the algorithm, it could be rather small but would grow with larger-scale problems with more levels.

At this stage in the research, it was very important to learn how to use a debugger. Using SUN Workshop, I was able to carefully track the algorithm in such a way that I was convinced that the interpolation and anterpolation routines were operating correctly. I used a small number of samples in order to track the code especially around the poles. The interpolation is necessary to match higher resolution radiation or receiving patterns at upper tree levels where the box size is larger. Larger boxes require richer patterns so the number of samples is increased from lower levels with fewer samples. By increasing the order of the interpolation, more neighboring points are used to find the new sample points at the higher level. Using at least fourth order Lagrange interpolation produced little differences in the interpolated data. To further test this and other parts of the code, it was necessary to develop a problem with very few unknowns.

6.5 Small-Scale Problems

In order to track the details of a code designed to solve large-scale problems, small test cases are necessary. I designed a two-particle test case with a separation of 64λ which allowed for many levels. Using this target, the difference in output RCS could be compared as a function of the number of levels and the interpolation order. I found that the RCS remained constant even with different numbers of levels used for a reasonable interpolation order. However, since RCS is an integration function that tends to average the effect of multiple scattering sources, solely relying on the RCS data to determine if interpolation was being performed correctly was not enough.

Digging deeper into the code, I studied the fine details on the interpolation process and concluded that it was a sound process. Furthermore, FISC was used for comparison, which led to the isolation of several differences. The truncation number used in the diagonalized translation operator was reset to be the same in both codes. In this way, both codes were set to match each other, but the resulting RCS values were still different. The next step was to find where the differences were coming from.

By extracting the matrix elements of the problem, the differences became apparent. Since MLFMA is a matrix-free method, both LSSP and FISC had to be modified to output an equivalent matrix. Since they produce matrix-vector products, by feeding the matrix vector product with a natural basis vector, the equivalent matrix can be found one column at a time. When the self terms were compared between the two codes, dramatic differences were noted. The self terms are computed using the method of moments and integration over the triangular basis functions.

Self terms and near-neighbor terms are integrated using more points as a rule. For example, if single point integrations are performed, the center point of each triangle is used in the computation for sufficiently separated basis functions while these self and near-neighbor terms use the next higher rule (in this case a four-point rule). Using a debugger to step through the algorithm line-by-line, I found that the points being used and the weights being applied were different between FISC and LSSP. FISC used standard Gaussian quadrature points on each triangle with the appropriate weights while LSSP used a crude integration rule with uniform weights. This crude four-point integration rule was a product of an early version of TRIMOM code.

Figure 6.2 shows the points selected by the early version of TRIMOM as depicted by the circles. The dark squares are the Gaussian quadrature points which have different weights. A full discussion of Gaussian quadrature integration on triangles can be found in [19].

The four-point integration rule was changed and checked to verify that points were chosen in FISC and LSSP in the same locations and with the proper weights. This resulted in self-term matrix elements being the same between the two codes. With this fixed, I was very excited to test LSSP on a large, 5-million-unknown sphere problem. I fully anticipated that this was the bug that was causing the extra

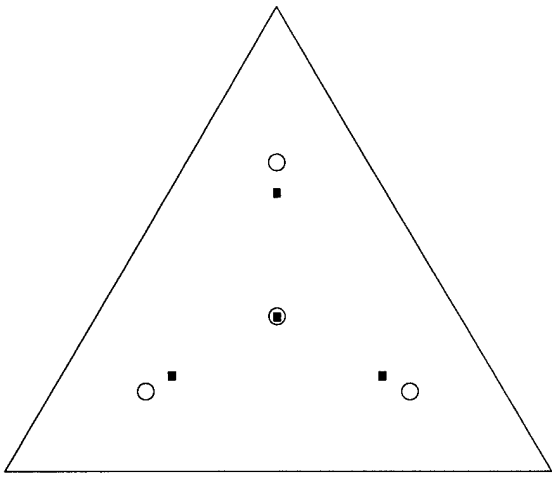

Figure 6.2 Four point integration rule. The circles are the original points chosen by TRIMOM and the squares are the Gaussian quadrature points.

noise problems in large-scale problems. Although the mean square error improved, to my disappointment, the problem persisted; however, I had learned and deepened my understanding of how these triangular basis functions are integrated.

The next step was unclear but required the output of a full unpreconditioned matrix. When this was done it seemed that the off-diagonal terms had problems with their signs. The problem seemed random and was quite disturbing. It led to extensive probing of the code and checking of the indexing and ordering of the basis functions and their points. Finally, through a discussion with Dr. Jiming Song, he suggested that it could simply be a different convention in choosing the ordering of the nodes. He said that it would be accounted for in the right-hand-side vector. I checked this and found his hypothesis to be true.

Next, FISC was modified to use Gauss-Lobatto integration points over the unit sphere. In Gauss-Lobatto, the poles of the sphere are used as points. This requires $2L^2+4$ points using the poles instead of $2L^2+2L$ Gauss-Legendre points. By using these points instead of the Guassian-Legendre points, the matrix elements between FISC and LSSP should be the same. Of course, these integration rules produce very similar results, but remember that I was searching for minor differences producing the noise observed. The matrix elements for small test-case problems were different. The difference was only found in the off-diagonal terms. Differences were noted whether the basis functions were close and done through MoM or far and done

using MLFMA.

At this stage of the debugging process, I had inserted a minus sign error into a subroutine to make it look more like FISC. It turned out that this affected the phase in an inconsistent way and produced matrix elements that were different depending on how many levels were used. This caused me to explore the order of the interpolation used between levels. By changing the interpolation order higher and higher, the effective matrix elements determined through matrix-vector-product testing should converge with higher-order interpolation. In FISC, the number of levels used for a two-particle test case produced nearly the same matrix elements, whereas LSSP elements were a different story. Luckily, I was using the concurrent versions system in Unix called CVS to control the code versions and configurations. Using the utilities to find code changes, this self-introduced problem was easily found and corrected. This was not the first time CVS had saved me in the debugging process. This is a key lesson learned in debugging. It is imperative to keep a history of changes made in a code. This is particularly applicable when the code is large and has many subroutines that interact together.

After checking the interpolation process again and verifying that the impact was small when changing from fourth to sixth order interpolation, this part of the algorithm was again exonerated. For quite some time, it was suspected that interpolation was not implemented correctly in ScaleME. This is not the case. At this time, the RCS computation was considered and studied to determine if errors were creeping in after the solution vector was found. This part of the computation appeared to be correct. By the end of March 2003, Dr. Velamparambil suggested that a complex bug involving the near interactions and more than eight multiple processors was the problem. He thought that it might be an indexing problem with multiple processors. I verified that the matrix-vector-product was slightly different when using 16 or more processors as compared to doing the problem on 1, 2, 4, or 8 processors. The error was in the fourth or fifth decimal place as opposed to the sixth or seventh decimal place with fewer processors. Dr. Velamparambil is still exploring this problem, but it was not the one responsible for the error we were looking for.

The next test involved a line of triangles. In this test, each RWG basis function shared at least one common triangle with the neighboring RWG basis function. Inside basis functions shared both triangles with neighboring ones. This was a

good test for checking effective matrix elements based on different distances from each other. The distances determine whether singularity extraction is required or whether MLFMA will be used to calculate the interaction. During this testing I found that some of the differences in the matrix elements between FISC and LSSP were due to a difference in the criteria for singularity extraction. FISC used a distance parameter based on the average edge lengths while LSSP used a frequency dependent distance. Clearly these choices are related but are not equivalent. The relationship is tied to the fact that edge lengths are set based on frequency and usually use 5 to 10 edges per wavelength.

6.6 Bug Annihilation

After all of these small-scale problems and testing, the final bug was found. In April 2003, after discussing all of the possibilities and without being able to align the LSSP-produced MLFMA equivalent matrix terms with those produced by FISC, a constant in ScaleME was changed to align with the constants used in the TRIMOM code. The wavenumber constant used by ScaleME, which produces the interaction of the far terms, was based on the speed of light. Originally, in the middle of 2002, I had changed this from 20.9582 to 20.958 450 22 while carefully probing the code line-by-line. This term is $2\pi \times 10^9/c$. Since I could not find out what values were used to get 20.9582, I used c=299 792 458 m/s and a precise value for π. This made little difference in the results so I didn't pursue this issue in the summer of 2002. Ironically, if I had used c=300 000 000 m/s, I would have immediately noticed a dramatic improvement in the bistatic RCS scattered from a sphere as compared to the Mie series. Even though this value of the speed of light is approximate, it is consistent with the parameters used in TRIMOM.

With a change of this wavenumber parameter to 20.943 951 023 9, ScaleME-produced interactions that were consistent with TRIMOM-produced interactions. The essential difference was a fourth-digit discrepancy in the speed of light used to calculate the far interactions versus the near interactions. This was the bug that produced the noisy data, which worsened as the problem size increased.

In order to give some physical interpretation of this error, one must consider the formulation and mathematics surrounding MLFMA. The extinction theorem, which essentially states that the internal fields are zero, is based on a single Green's

function. This Green's function is used to find the surface currents necessary to completely cancel the incident wave for any point inside the enclosed region [14]. With two different Green's functions being used, the internal fields may not be identically zero everywhere. Internal fields would produce the effect observed in the original data. We note that the one-way propagation across the diameter of a 200λ sphere will differ in phase by 0.87 rad by using the approximate versus the exact value for the speed of light. The surface wave would travel further to reach the same point and would have both an amplitude error greater than 3% and similar phase difference as the direct path. These two factors, which result from inconsistent Green's functions, cause the formulation to produce the undesirable results. A physical explanation does not exist as this is simply a mistake in the formulation with conflicting Green's functions.

The first target that I verified the newly compiled code on was a sphere with 5 million unknowns. The newly produced data was beautiful and clearly conveyed the fact that the bug had been annihilated. Figure 6.3 shows this bistatic RCS data before the bug fix, and Figure 6.4 shows this bistatic RCS data after the bug fix. Notice the decrease in the noise across the plot of data.

6.7 Summary

It is natural to contemplate what would have happened if I did not find the bug or found it at a different time. Ironically, I came in contact with this bug early during my time on the ScaleME project. I even changed this particular line in the code. What if I had uncovered this bug in the beginning? How would that have affected my research and personal technical growth? Many things were learned solely as a result of searching for this bug. Some of these details that are now clear may have remained fuzzy in my understanding without the depth achieved while digging into this error. On the other hand, perhaps I could have made different contributions or tested more targets had this bug been found earlier. It is interesting to consider the issues surrounding debugging codes.

The properties of this bug were such that a solution could have easily eluded me, as it did previous researchers. When I consider this possibility, I am grateful for the persistence and drive that I maintained throughout the process and the fact that I found success after so many long hours. It is very satisfying to find such an

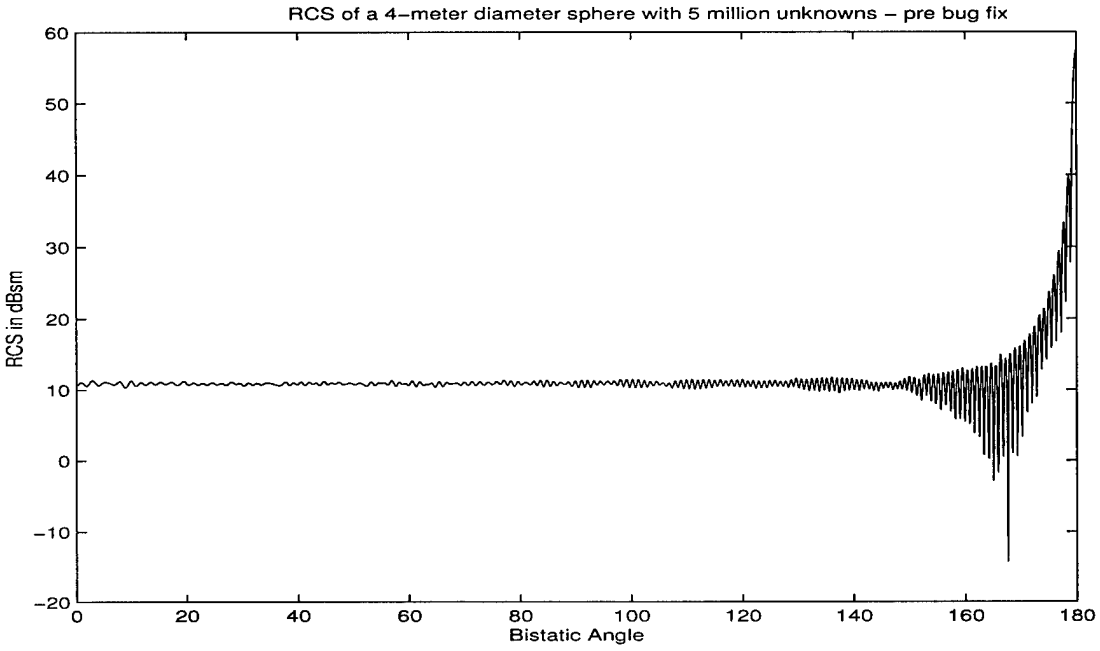

Figure 6.3 LSSP 5 million unknowns – prebug fix for a sphere.

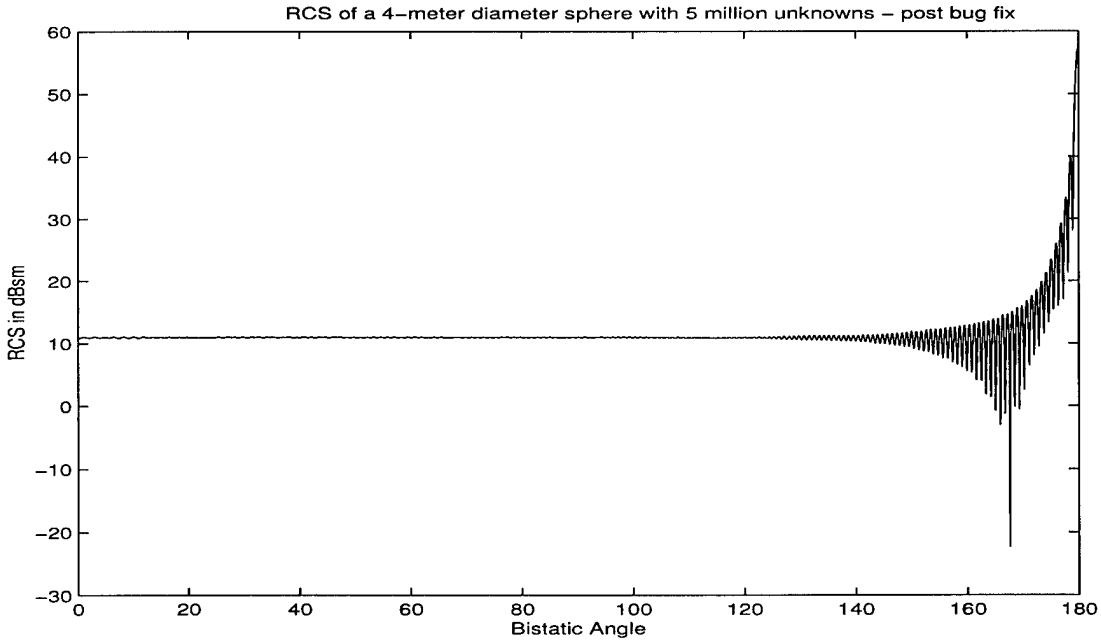

Figure 6.4 LSSP 5 million unknowns – postbug fix for a sphere.

illusive bug; however, not finding the bug would have been mentally crushing. The feeling would be like investing lots of money over a long period of time and having the investment go 'belly up' when you needed to use the money. It is noteworthy to mention that this outcome was a likely possibility and a reality that some programmers or researchers must face. I am very pleased to have worked in the environment of the Center for Computational Electromagnetics at the University of Illinois where the bug could be isolated, found, and annihilated.

To conclude this chapter documenting my research journey in a key part of computational electromagnetics, namely debugging, I present the following plots in Figure 6.5. These four graphs show the same target sets as given in Figure 6.1 with the error fixed and the same plot scales for comparison. The variation is now at an acceptable level compared to the previous data plagued by growing noise.

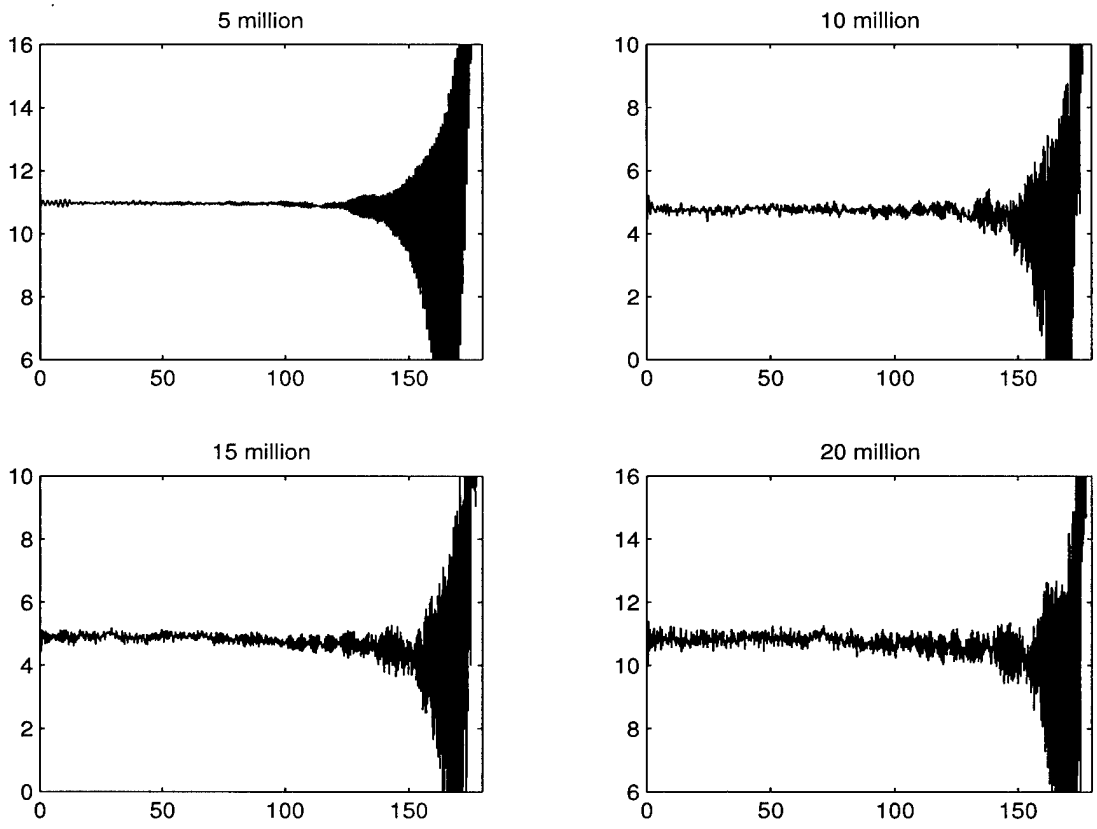

Figure 6.5 Large-scale sphere error after bug fix.

CHAPTER 7

PARALLEL EFFICIENCY AND SCALING

7.1 Introduction

To solve truly large scale problems in a reasonable time requires parallel processing. Ideally, one would want computations to finish X times faster on X processors; however, more processors imply more communication. This communication overhead decreases the efficiency such that the speed will be less than X times faster. Also, unequal division of labor and poor load balancing over individual processors lead to a further decrease in speed performance. Finally, some parts of the process require a serial approach in which certain data or information must be calculated or completed prior to moving to the next step. All of these issues affect the efficiency of an algorithm when employing multiple processor machines.

Scalability is the ability for an algorithm or process to maintain functionality as the problem size increases or the number of parallel nodes increases. Functionality is defined according to a desired efficiency. For example, if a problem takes 10 h with 1 processor and only 1 h with 10 processors, it has perfect efficiency and obviously scales for 10 processors. If the same problem is solved using more than 10 processors and still yields a solution in 1 h, one might say that the process does not scale beyond 10 processors. It is more likely that a scalable process will have a decreasing efficiency with an increasing number of processors. Eventually, the problem will become inefficient and may take even longer on a greater number of processors. A good analogy is to compare the idea of scalability to the manufacturing process. Adding more workers may at first have a linear effect on the work production; however, at some point when too many workers are added, the structure may give rise to some workers not doing anything to contribute to production, or worse, hindering the production of others.

This chapter will discuss the early efforts to create a parallel MLFMA and the success of this parallel algorithm created in the code ScaleME (Scalable Multipole Engine). Initially, MLFMA was closely tied to the method of moments algorithm in

a way that made it efficient on a single processor but difficult to parallelize. Using shared memory machines, the loops can be parallelized in FISC but the scalability beyond 16 processors is inefficient. LSSP, on the other hand, uses ScaleME to speed up the matrix-vector product while allowing for an independent MoM code to be employed. ScaleME was designed for parallel implementation from the start. Overcoming the challenges was a process that evolved over four years of hard work [20, 21]. With all things equal, the memory requirements on a single processor were the same as FISC while the speed was 23% slower due to optimizations in FISC that were not implemented in the ScaleME project [5].

Ideally, one would like to use the individual memory banks on each computer node in such a way that the memory required for the problem can be equally divided among the nodes. This kind of memory allocation is called distributed memory and is the opposite of shared memory. In other words, each node can access its own memory but not the memory of the other nodes. This leads to the requirement of message passing when nodes need to communicate information necessary to solve the entire problem. Message passing requires special software and techniques to be effective. Careful consideration of the message lengths and synchronization in sharing information is required. The extra communication time leads to run time delays. Also, the workload needs to be equally divided in a way that allows each processor to be fully engaged at all times without having to wait too long for information to be received from other nodes. Since parallel implementation generally requires dedicated networks on supercomputers, it is important to be able to estimate both memory and run times before submitting a job to a batch queue.

7.2 Memory

Huge amounts of memory become necessary as the problem size grows. The memory requirements for MLFMA are $O(N \log N)$ and have been estimated through empirical methods [5, 22]. On a single processor, this estimate is good but does not apply when multiple processors are employed. There are several parts of the memory requirement that must be replicated on each processor for efficient calculations. The parameter settings have a significant effect on the memory requirements. This section will discuss the settings used in LSSP and ways to estimate the memory

a priori. Since the amount of memory required is dependent on so many parameters, including target geometry, in order to achieve accurate memory estimation on large-scale targets, it may be better to make some test runs first. There are several techniques to initially test the target. Memory estimation is key to knowing the boundaries on problem sizes.

The memory requirements stem from the following sources: near interactions, radiation and receiving patterns, translation operators, shifting matrices, preconditioning, box patterns, solution work arrays, and geometry. Some of these stay constant as the number of processors is increased; however, others increase as the number of processors increases. The tree structure can play an important role in the memory usage as will be seen in some of the forthcoming subsections detailing memory estimation. In MLFMA, the highest-level translations occur on level 2. Because the box size at this level is the largest, it requires a lot of samples for the radiation pattern based on the excess bandwidth formula of Equation (4.4). In other words, more modes are needed to describe the richness of the pattern at the topmost level. Many fewer samples are required at the finer, lowest levels.

The number of boxes filled with target geometry information at each level must be known or estimated to predict the amount of memory required. At each subordinate level in 3-D, oct-tree, there are eight times the number of boxes in the higher level. At level 2 there are 64 boxes. Typically, we have found that 4 out of the 8 boxes are filled at each level. Depending on the general shape of the target, the number of filled boxes at level 2 could cover a broad range. For example, a narrow cylinder offset from the origin could fill as few as 4 boxes while a sphere will fill 56 boxes out of the possible 64 boxes. Taking the case of the sphere, it is possible to predict the number of filled boxes at each level using the seed number of 56 filled boxes on level 2 as seen in Table 7.1.

Knowing the tree structure and the number of filled boxes at each level is essential for accurate memory estimation. The size of the box at each level can be predetermined. Using these box sizes, the truncation number can be selected based on the excess bandwidth formula or a method described in Section 4.5.1. This truncation number will lead to a certain number of samples at each level. The number of samples at the lowest level is probably the most important factor in memory usage. Table 7.2 shows the number of samples for a 10-level sphere target of 20 million unknowns.

Table 7.1 Actual versus estimated filled boxes for memory estimation. The number of actual filled boxes for a sphere with 20 million unknowns is similar to the predicted number. The estimate is based on a factor of four at each successive level using the level 2 seed of 56. The estimate could be improved if a level 3 seed of 272 were known *a priori.*

Level	Actual	Estimated
2	56	56
3	272	224
4	1160	896
5	4760	3584
6	19 176	14 336
7	76 085	57 344
8	297 045	229 376
9	1 138 787	917 504
10	4 180 986	3 670 016

Table 7.2 Actual filled boxes and number of samples for memory estimation. The number of actual filled boxes for a sphere with 20 million unknowns is shown with the number of samples at each level. For the lowest level, 36 samples are needed which is based on a truncation number $L = 4$ and $2L^2 + 4$.

Level	Actual	Samples
2	56	640 716
3	272	167 046
4	1160	44 406
5	4760	12 172
6	19 176	3366
7	76 085	972
8	297 045	292
9	1 138 787	102
10	4 180 986	36

Using information like that found in Table 7.2, the different memory requirements can be estimated. The next several sections will individually discuss how to estimate these requirements based on the run parameters. The example of this 20 million sphere will be used so this table will serve as a good reference.

7.2.1 Radiation patterns

The radiation patterns are associated with the number of samples at the finest level and the number of unknowns. We need a θ and ϕ component for each box sample. This is multiplied by the number of unknowns and requires 8 bytes of storage. In this example we need $36 \cdot (20 \cdot 10^6) \cdot 2 \cdot 8$ bytes or 10.7 Gbytes. If the truncation had been $L = 5$, the number of samples would have been 54 instead of 36 and so this memory requirement would have increased by 50%. To achieve higher accuracy at the lower levels, the truncation number is usually increased. The radiation patterns are independent of the number of processors being used.

7.2.2 Translation matrices

In order to perform translations at each level, memory is required for each possible translation direction. Symmetry can be used to decrease the memory or compressed translators can be computed on the fly. The memory for translation does depend on the number of processors, as the translators must be available on each node. In the case of shared levels, a specified number of coarse upper levels are shared between processors. The translators for these levels do not require replication at each node. To calculate the memory for uncompressed translators, we need to know the number of possible translation directions. This is found by considering the unique directions and distances of each of the eight child boxes in a parent cube. Since the translations of a child cube are always to near neighbors of the parent cube, the total number is $7^3 - 3^3 = 316$. Using this number and the number of samples at each level, the memory can be calculated. We need 8 bytes per sample. In simple equation form, the memory is

$$\mathbf{Mem_{O2I}} = 8 \cdot 316 \cdot \left[\sum_{l=2}^{S} N_{s_l} + N_p \sum_{l=S+1}^{\textbf{max level}} N_{s_l} \right] \tag{7.1}$$

where S is the maximum shared level, N_{s_l} is the number of samples at the l-th level, and N_p is the total number of processors. If shared levels are not used, the first summation is zero and the second begins with $l = 2$. Using this equation

Table 7.3 Translation memory estimation. The memory for the translation operator is dependent on the number of processors and shared levels. Using compressed translators decreases this memory. The advantage of shared levels is easily seen.

Memory in Gbytes	Shared Levels				
Processors	0	2	3	4	5
8	16.4	5.8	3.1	2.3	2.1
16	32.7	10.1	4.2	2.6	2.2
32	65.4	18.7	6.5	3.3	2.4
64	131.0	35.9	11.1	4.6	2.8

and the number of samples for our 20-million-unknowns example, we can produce a table of memory requirements based on shared levels and number of processors (see Table 7.3).

7.2.3 Near neighbor interactions

The memory for near neighbor interactions is the most difficult to estimate. The number of levels used has a dramatic impact on this memory. For example, by using one level fewer, this memory will increase by a factor of four. We have found that this memory is related to the average number of unknowns in the finest level boxes. If every filled box had this average number of unknowns and each filled box had a fixed number of near neighbors, the near neighbor interaction memory could be estimated precisely. However, both these quantities vary with box size and geometry. Nevertheless, we can figure the one-way interaction of each filled box and its neighbors by using the average number of particles, N_{avg}, and an estimated number of near neighbors. If we assume, as before, that half of the 27 blocks in a near neighbor interaction cube are filled, then this memory for all of the filled boxes will be

$$\mathbf{Mem_{near}} = 8 \cdot N_{avg}^2 \left[\frac{27}{2}\right] \cdot N_f \tag{7.2}$$

where N_f is the number of filled boxes at the lowest level and $N_{avg} = N/N_f$. If we replace N_{avg} with the number of unknowns divided by the number of filled boxes, the memory for near interactions can be written as

$$\mathbf{Mem_{near}} = 8 \cdot \left[\frac{N^2}{N_f}\right] \left[\frac{27}{2}\right]. \tag{7.3}$$

In this form, the dependence on N^2 appears ominous but with a lot of levels N_f is also a very large number. We can also see that since N_f at the next coarser level is different by a factor of four, the memory for one less level will increase by this same factor. The near interaction memory is geometry dependent, so this estimate only serves as a guide to the actual amount of memory required. Although this is a reasonable estimate most of the time, the factor of 27/2 can be adjusted between 27/3 and 27/1.5. These correspond to 1/3 and 2/3 of the boxes being filled. In the 20-million-unknowns example, the near-interaction memory was estimated at 9.62 Gbytes which was approximately the same as the actual 9.57 Gbytes used.

7.2.4 Box patterns

Memory for box patterns is required at each level for the upward pass and for the downward pass. In theory, the memory used in the upward pass can be recycled on the downward pass, but freeing memory is very machine and compiler dependent. The memory required at each level of a pass is 8 bytes times the product of the number of filled boxes and the number of samples required on that level, which is then multiplied by two for both θ and ϕ components. If we consider both the upward pass and downward pass without any memory recycling, we get the relationship

$$\mathbf{Mem_{boxes}} = 2 \cdot 2 \cdot 8 \cdot \sum_{l=2}^{\text{max level}} N_{s_l} \cdot N_{f_l} \tag{7.4}$$

where N_{f_l} is the number of filled boxes at the l-th level. The memory for the box patterns tends to increase from the coarsest level to the finest levels. The finest three levels in our 20 million unknowns example account for half of the box pattern memory. In this case the bottom third of the levels represent half the memory.

7.2.5 Block diagonal preconditioner

We use the block diagonal preconditioner as a quick way to prepare the matrix-vector product solution to require fewer iterations. The basic idea of this preconditioner is to take the self-interaction terms of a given cube and invert the associated matrix. The matrix associated with each box at the finest level has N_{avg}^2 terms. Therefore, the block preconditioning will require

$$\mathbf{Mem_{bdp}} = 8 \cdot N_{avg}^2 \cdot N_f. \tag{7.5}$$

Since N_{avg} is a decimal number, we have found that rounding it up produces a more accurate estimate. Once again, N_f is the number of filled boxes at the lowest level. The $\mathbf{Mem_{bdp}}$ will tend to underpredict the memory required for the block diagonal preconditioning. Rounding up to an average of five basis functions per filled box will result in a predicted memory of 797 Mbytes compared to the actual 980 Mbytes needed. It may be reasonable to add one to N_{avg} before it is squared to try to get closer to the memory needed. In this case, adding one to N_{avg} would make the estimated memory 1067 Mbytes, which is a reasonable overshoot.

7.2.6 GMRES

The iterative solution requires memory whose quantity is associated with the number of unknowns and the number of restarts. Generalized minimum residual (GMRES) uses the previous solution vectors to choose an orthogonal vector for the next iteration. Because the number of iterations could make this a huge memory requirement, GMRES allows for the solver to restart and clear the memory used in the previous iterations. This memory is easy to accurately predict as

$$\mathbf{Mem_{GMRES}} = 8 \cdot N \cdot (5 + N_r) \tag{7.6}$$

where N is the number of unknowns and N_r is the number before restart. In the example of the 20 million sphere, this was 2.2 Gbytes using 10 as the restart number. The predicted memory matched the actual memory.

7.2.7 Geometry

The memory required for the geometry information includes memory for nodes, facet connectivity data, and edges. This has a modest dependence on the number of processors because some parts of the geometry must be replicated in duplicate processors. In the early days, the geometry was replicated on each processor but for large problems this was not feasible. Now, the geometry data is load balanced to all of the processors fairly equally. To estimate the total memory requirement in bytes, we multiply the number of unknowns by 60. For the 20-million-unknowns case, the geometry represented 1.12 Gbytes. On many processors, this will still be within 10% of the actual memory predicted including overhead.

7.2.8 Total memory

Summing these major contributors to find the total memory, we see that the 20 million sphere required 48.1 Gbytes which was spread across 32 processes. The current implementation of LSSP only allows for uniform load balancing and a maximum memory of any single process of 2 Gbytes. This is due to the message passing interface (MPI) compilers. This large-scale sphere problem was solved on 10 dual processor nodes of the local SUN Blade 2000 cluster. It required 32 processes on 19 processors to optimize the memory usage of each node. Different amounts of memory were available on each node so some nodes were given four processes and other nodes were given three processes. The master node only used one processor. The average process required 1.5 Gbytes of memory. Since the processors had nonuniform workloads, some additional waiting was required to allow the heavier loaded processors to catch up and pass the required data to the other waiting processes. Estimating the memory is one important area in conducting performance studies of LSSP, but estimating the solution time is still very hard.

7.3 Time

Estimating the solution time requires more research. There are many factors that contribute to the solution time. Processor speed, of course, plays a big role. Since MPI is used to pass information between processes, the network speed can play a significant role in parallel MLFMA. The run time can also be decreased through native compiler optimizations. The memory and time used by FISC is given in [22] and is comparable on a single processor to LSSP [5].

The time can be split into two categories: setup and solve. The setup time is based on the surface area of the geometry, which is related to the number of unknowns. Through load balancing, the geometry can be parsed out to different processors in an equal fashion. The translation matrices are established and the near interactions are computed during this phase. The second phase is the time to solve. This is based on the time required per matrix-vector product. Since MLFMA is used, the time is $O(N \log N)$. With multiple processors employed, the time is based on the parallel efficiency. The parallel efficiency decreases as the number of processors increases for a fixed number of unknowns. However, if the

number of unknowns is increased in order to solve a particular geometry at a higher frequency, the number of processors can be proportionally increased without loss of efficiency. This is the principle of scaling.

7.4 Parallel Efficiency and Scaling

One of the goals of this research was to study the performance of the code LSSP. Hundreds of runs have been made on different targets. Target geometries include spheres, cylinders, and complex targets like aircraft. When testing the parallel efficiency, the target is run on a single processor to establish the time per matrix-vector product. The target is then run with the same input parameters on 2, 4, 8, 16, ... processors. The parallel efficiency of the matrix-vector product can then be determined using the ratio of the single processor time to the product of the number of processors and the associated runtime. For example, if a matrix-vector product on a single processor takes 100 seconds and on two processors it takes 52 s, the ratio $100/(52 \cdot 2)$ is 96%.

Although a sphere is a great test object, most real world objects lack 3-D symmetry and tend to have some sharper edges and corners. For this reason a set of benchmark targets were created that might represent many basic target shapes. Three cylinders of equivalent surface area were designed such that the ratio of circumferences to lengths were varied according to Table 7.4. Then each target was remeshed into finer edges on the order of 125K, 500K, and 2M. In this way, the targets could be compared for parallel efficiency. Because the surface areas are the same, each target, C1 to C3, was run at the same frequency corresponding to the same mesh refinement. The parallel efficiency was studied dependent on the number of shared levels used. Shared levels are described in [4] and play a significant role in parallel efficiency as the number of processors increases. They also help out in cutting down on the memory required for translation operators. The primary idea behind the shared levels is to take the upper level boxes and replicate them on each processor while splitting the far-field patterns equally over each processor. This reduces the need for message passing at the uppermost shared levels where the message lengths would be extremely large.

Data for the 500K cylinder set is given in Table 7.5. This table shows the parallel efficiency (in percent) of the matrix-vector product between 1 and 16 pro-

Table 7.4 Cylinder set: stick, can, nickel.

Cylinder	Nickname	Radius	Circumference	Length	C/L
C1	stick	0.34	2.12	21.2	0.1
C2	can	1.00	6.28	6.28	1.0
C3	nickel	2.11	13.3	1.33	10.0

Table 7.5 Parallel efficiency of the stick, can, and nickel cylinders. Efficiency is given in percent. The comparison is between 0 and 2 shared levels. For these cylinders with about 500K edges, the shared levels do not appear to enhance performance.

Processors	Stick		Can		Nickel	
Shared Levels →	0	2	0	2	0	2
1	100	100	100	100	100	100
2	89.2	82.9	94.6	100	103	98.4
4	85.1	78.3	89.1	94.5	100	99.4
8	85.0	70.9	97.0	97.0	98.4	99.3
16	70.0	69.3	91.2	87.0	101	94.7

cessors comparing no shared levels and the highest level 2 being shared. Data points above 100% imply that the run finished faster than the single processor run divided by the number of processors being compared. This may be due to computer cache memory 'hit and miss' and is not an important point. The key observation is that shared levels do not appear to enhance the parallel efficiency of LSSP for this problem size. Other tests have revealed dramatic improvements in parallel efficiency through the use of carefully chosen shared levels. Additional data needs to be collected and processed to better understand the performance.

The parallel efficiencies for cylinders C2 and C3 are given in Table 7.6 comparing shared levels 3 and 4. From this we see that the parallel efficiency is very good up to 32 processors and is reasonable up to 64 processors. Also, this data shows that increasing the number of shared levels from 3 to 4 does not necessarily increase the parallel efficiency. However, in Table 7.7, the shared levels have a dramatic effect on the parallel efficiency of the pencil target. From this data, it

Table 7.6 Parallel efficiency of the can and nickel cylinders for 3 and 4 shared levels. Efficiency is given in percent. The comparison is between 3 and 4 shared levels. For these cylinders with about 2 million edges, the shared levels help enhance performance.

Processors	Can		Nickel	
Shared Levels →	3	4	3	4
1	100	100	100	100
2	92.1	89.7	102	94.4
4	96.9	79.4	99.7	90.9
8	104	83.8	103	88.2
16	97.7	98.3	92.1	79.8
32	84.3	84.3	92.9	84.3
64	61.5	72.1	63.6	66.0

Table 7.7 Parallel efficiency of the pencil target. This efficiency illustrates the improvement associated with increased shared levels.

Processors	Pencil			
Shared Levels →	0	2	3	4
1	100	100	100	100
2	90.7	88.4	99.5	102
4	77.5	82.0	97.0	98.0
8	66.0	76.2	95.6	95.6
16	42.2	60.4	93.3	93.6

appears that the efficiency improves with more shared levels.

The pencil, stick, can, and nickel data were all run on the SGI Origin 2000 using the local NCSA supercomputing system here at the University of Illinois. It is hard to find nonsupercomputing resources that have up to 64 processors for such testing. More data is required to fully study the scaling properties of LSSP. With the newly acquired ability to predict the memory, such testing should be much easier.

CHAPTER 8

LARGE-SCALE PROBLEMS

8.1 Introduction

This chapter shows the results of large-scale problems. By large scale, we mean the solutions of millions of unknowns or targets greater than 100 wavelengths. At the present time, the solutions of spheres with 10, 12, 15, and 20 million unknowns are shown. The path up to 10 million was accomplished by previous researchers [5, 20]. This chapter will revisit that point and show recent heights. Before showing these large-scale problems, a discussion of run optimization will be given.

8.2 Run Optimization

With an understanding of memory given in Chapter 7, it is possible to consider ways to optimize a given run. Normally a geometry model is available and a set of required frequencies that need simulations are given. These frequencies are used to calculate the average edge length of the target in terms of wavelengths. Typical discretization is between 0.1λ and 0.2λ edge lengths. If the average edge length is outside of this region, it will probably be necessary to remesh the target to get within good modeling standards. At the present time, we use an automatic remesher in FISC that determines which edges are too long based on input criteria. If an edge is too long, it is divided into two. Therefore, depending on how many edges of a triangle need to be divided, a triangle could produce up to four triangles. If the edges of these triangles are too long, then they are subdivided until all triangles meet the expected criteria. Figure 8.1 shows the three different ways that triangles are subdivided. After one of these three subdivision is made, the newly produced triangles are evaluated for sides that exceed the maximum dimension and further subdivisions may be made if necessary.

If the edges are roughly uniform in length, subdivisions will produce meshes with four times the number of triangles and edges. This means that the average edge length and the number of unknowns is hard to control. It would be possible

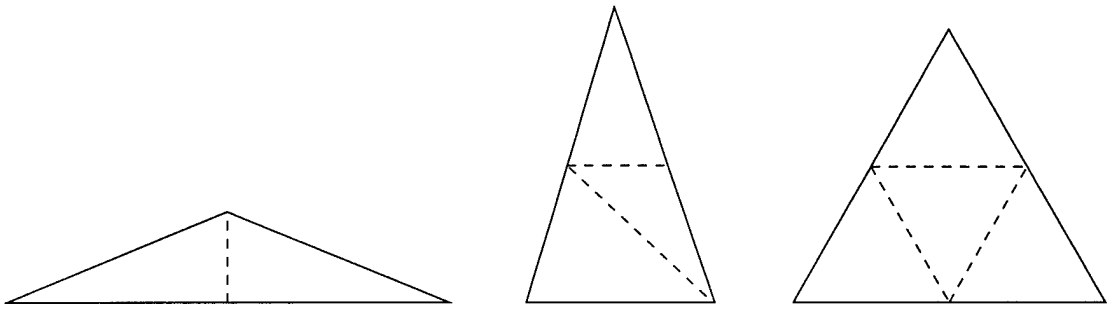

Figure 8.1 Triangle facet subdivision. The example on the left has one edge that exceeds the maximum edge length while the one in the middle has two edges that exceed the length. The triangle on the right has all sides that are too large and require subdivison.

to remesh a target with a modeling program rather than this simple scheme, but precision control of edge lengths and edge quantities is difficult to manage. It is important to understand the relationship between the number of unknowns and the surface area of a target and how it relates to this average edge length. Assuming a mesh created with equilateral triangles, the minimum average edge length can be found based on the total surface area and number of triangular patches called facets. The relationship between the area of a single facet and its surface area is easily derived as

$$A_f = \frac{\sqrt{3}L_e^2}{4} \tag{8.1}$$

where A_f is the area of a facet and L_e is the length of an edge. The surface area A_s of this target is found by multiplying the number of facets N_f by the area of the average facet A_f. Since the number of edges N is the number of unknowns in a problem and is related to N_f by a factor of 1.5, the relationship between N and L_e is given in terms of a fixed target surface area as

$$N = \frac{2\sqrt{3}A_s}{L_e^2}. \tag{8.2}$$

From this equation we can see that the number of edges is inversely proportional to the square of the average edge lengths. For a fixed edge length, the number of unknowns will increase proportionally to the surface area. When the triangles are not equilateral, the average edge length increases over what this equation predicts. This occurs because the proportion of surface area to the number of unknowns

decreases. For example, the VFY-218 with 9 990 918 edges that was run at 8 GHz had a surface area of 162.8 m^2, which is equivalent to 115 789λ^2 at this frequency. The average edge length was 0.2046λ. Using this average edge length and the appropriate surface area in Equation (8.2), the expected value of N for the number of unknowns would we 9 581 790 which is only about 4% lower than the actual value. This indicates that most of the triangles are quasi-equilateral. Without actually reading the edges into memory and computing the average lengths, this same equation can be solved for the average edge length as

$$L_e = \sqrt{\frac{2\sqrt{3}A_s}{N}}. \tag{8.3}$$

Then, by using this equation, the average edge lengths for a geometry can be estimated prior to deciding on the optimal way to run the target. In the example of the VFY-218 with nearly 10 million unknowns, the calculated average edge length based on the actual surface area and the actual number of unknowns is 0.2004λ. The actual average is higher as expected.

With this background information, it is possible to optimize any particular run. A description of the process is given below. In an example, we assume that the target surface area is given or known and data is required at a particular frequency. A finer mesh is desirable for accuracy but may not be practical for a large-scale problem. Assume that the target is physically large with a surface area of 200 . This target geometry is composed of 200 000 facets, or triangles, which means there will be 300 000 edges, or unknowns, to compute for this level of target discretization. If this were a sphere, it would be about 8 m in diameter. A fighter-size airplane might be 15 m long.

Using these two dimensions and a requirement to produce RCS data at 1.5, 3, 6, and 10 GHz we examine the procedures to optimize such runs. Equation (8.3) is plotted in Figure 8.2 for this 200-m^2 target and these four frequencies. First we consider the lowest frequency of 1.5 GHz. The average edge length of the original target with 300 000 unknowns is 0.048 m, which at this frequency is 0.24λ. Assuming all of the edges are this length, each triangle would be subdivided into four new triangles whose average edge lengths would be 0.12λ. Since we prefer to be in the range of 5 to 10 edges per wavelength, this fits perfectly. Also, because the frequency is in the lower range of those required, fewer unknowns are required

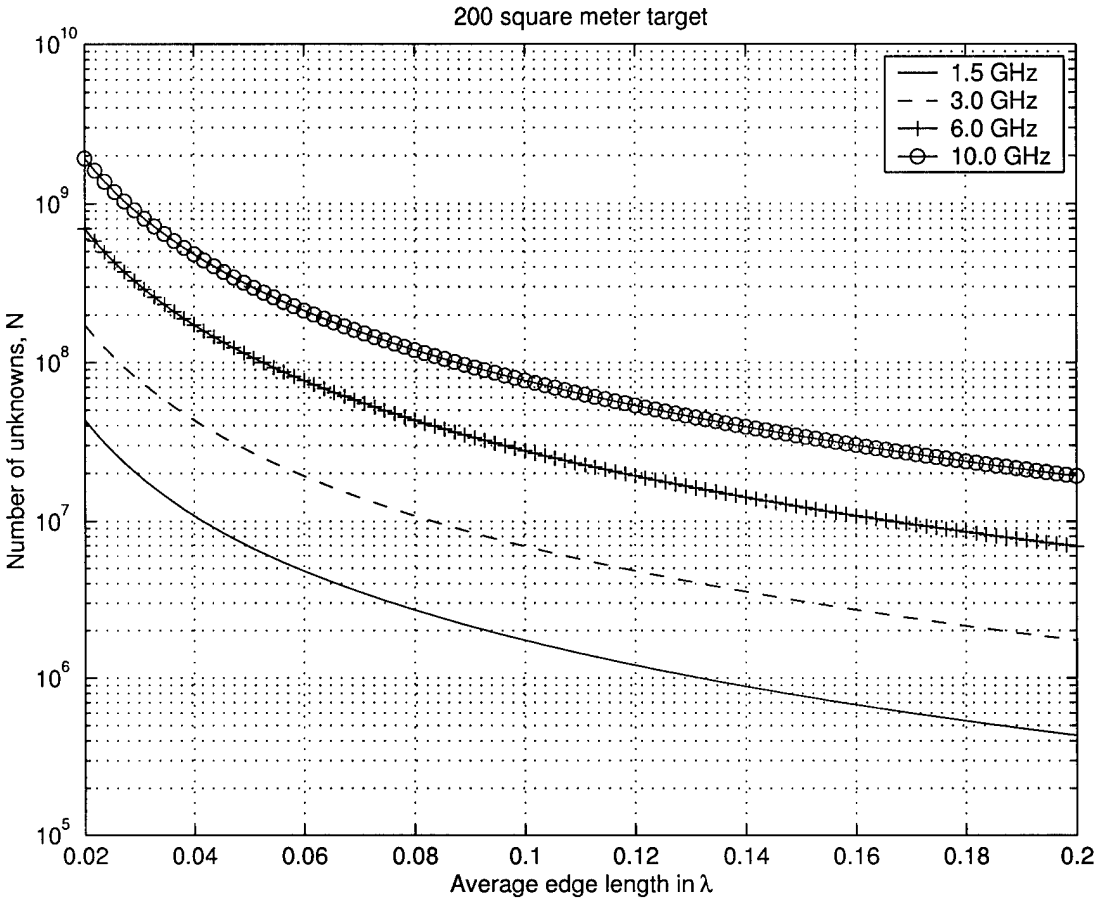

Figure 8.2 Number of unknowns compared to average edge length. For this target of 200 m^2, the number of unknowns for four different frequencies are shown based on different average edge lengths. In this example, at 10 GHz, 20 million unknowns would be required for 0.2λ while almost 80 million unknowns would be needed to represent the target with 10 edges per wavelength.

so this fairly fine discretization will work. At the higher frequencies, the number of unknowns may require minimization in order to fit the problem into available memory constraints.

Once the average edge length is determined and a finer discretization of the target is achieved, the minimum size of the lowest level box in the MLFMA tree can be found. It should be at least 1.5 times the average edge length. In this case, the lowest level box could be as small as 0.18λ. In the case of an 8-m-diameter sphere that can be contained in a 40λ cube, a 7-level MLFMA tree could be used with a lowest-level box with a length of 0.3125λ. Each level successively cuts down

Table 8.1 Samples required based on truncation number.

	Number of Samples	
L	Lobatto	Legendre
3	22	24
4	36	40
5	54	60
6	76	84
7	102	112
8	132	144
9	166	180

the size of the box dimension in half. If the target was an airplane 15 m long, we would expect to go down an additional level. At 1.5 GHz, this length would be 75λ and the 8-level tree would have a smallest box size of 0.2930λ.

With the number of levels determined, Chapter 7 can be used to determine how much memory will be needed based on the accuracy desired. One of the key parameters is the number of samples required at the lowest level. This is based on the truncation number described in Chapter 4. The smaller the box, the fewer the samples needed to represent the radiation pattern from sources within the box. One important issue is the discrete nature of the number of samples. Since the truncation number L is squared to arrive at the number of samples needed for each level, the jumps can be pretty significant. Table 8.1 shows how the number of samples increases based on this truncation number. Also, by increasing the number of levels, certain memory requirements are dramatically reduced. The number of processors to be used also has an impact on the memory. Since the program is highly scalable with a high parallel efficiency, many processors can be used to exploit distributed memory and to harness parallel processing speed.

To illustrate the same target within C-band (at 6 GHz), the lengths are 160λ for the sphere with 200 m^2 surface area and 300λ for an aircraft shape. Looking at Figure 8.2 at the line representing 6 GHz, we are targeting an average edge length between 0.1λ and 0.2λ, which corresponds to the number of unknowns somewhere between 7 and 28 million. Since these are ultra-large-scale problems,

fitting them into memory can be a problem so we might want to optimize for the coarser representation to cut down on the number of unknowns. Assuming we could produce a geometry with either endpoint number of unknowns, we can calculate the number of levels that could be used in either case. Using the 1.5 factor, the smallest boxes could be, respectively, 0.15λ and 0.3λ. Of course, 0.15λ is pretty small and may run into error problems because of its small size. For both of these discretizations, the target bounding box remains fixed. That is to say, for the sphere, a 10-level MLFMA tree would have a small box size of 0.1563λ and the box size for this many levels on the aircraft shape would be 0.2930λ. Of course, the first box size would be too small for a coarse discretization, so a 9-level tree would need to be used for a smallest box size twice this value, ie., 0.3125λ.

Using one fewer level will increase the memory for the near interactions by a factor of four. However, using more unknowns will increase the memory according to $O(N \log N)$. This is the tradeoff which requires careful analysis. Since the memory is based on digits of accuracy and many other factors, a single solution must be weighed against the available possibilities. Usually the ability to discretize the geometry to a certain level of fidelity drives a lot of the decisions, but the most important consideration is the ability to fit the problem into the available memory, which may be tricky for ultra-large-scale problems. Any problem beyond 10 million unknowns might be classified as an ultra-large-scale problem while problems between 1 and 10 million might be considered large-scale problems.

To attack ultra-large-scale problems when the amount of memory is a big issue, the accuracy parameters must be relaxed and using Table 8.1, steps should be taken to ensure the finest level boxes can minimize the number of samples needed at the lowest level. To illustrate the impact, consider the percentage decrease in memory for the radiation patterns when the truncation used changes from $L = 4$ to $L = 5$. The number of samples increases by 50%.

8.3 Large Spheres

The sphere is a classic problem and a great benchmark to study performance and accuracy. The results can be compared to the Mie series solution. Since the target is discretized, the solution will be slightly different but should compare favorably. To prepare to solve a sphere problem, a sphere mesh must be created. An

Table 8.2 Sphere edges. The integer used to create ultra-large-scale spheres with the corresponding number of edges.

Integer	Edges
913	10 002 828
998	11 952 048
1120	15 052 800
1291	20 000 172

automated program developed here at the Center for Computational Electromagnetics (CCEM) was modified to create spheres with more than 12 million edges, or unknowns. The program requires an integer input that is squared and multiplied by 12 to get the total number of edges. In Table 8.2, this integer is shown with the respective number of edges or unknowns.

The surface area of a sphere is $4\pi r^2$ and the surface is formed with triangular facets. There are twice as many facets as nodes and three times as many edges as nodes. Using these ratios we can determine the average edge length based on the radius of the sphere and the number of edges. By dividing the surface area by the number of facets, the average facet area is found. Assuming equilateral triangles, the facet area A_f is given by $A_f = L_e^2\sqrt{3}/4$, where L_e is the side or edge length. The average side length must be determined to set the frequency such that the edge length is about $\lambda/10$ to $\lambda/5$. The average edge length is finally given in terms of radius r and number of unknowns N as

$$L_e = r\sqrt{\frac{8\sqrt{3}\pi}{N}} \tag{8.4}$$

$$L_e = 6.6\frac{r}{\sqrt{N}} \tag{8.5}$$

and the frequency (in GHz) corresponding to an average edge length of $\lambda/10$ is $f_{\mathbf{GHz}} = 0.03/L_e$. In an easy to see form,

$$f_{\mathbf{GHz}} = \frac{\sqrt{N}}{22sr} \tag{8.6}$$

where s is the number of segments per wavelength, or the inverse of L_e, and is usually between 5 and 50. From this equation we can see that the frequency must

Table 8.3 Sphere frequencies. The frequency in GHz is given for the number of edges per wavelength s and the number of unknowns or edges assuming a one meter radius sphere.

	Edges			
s	10 002 828	11 952 048	15 052 800	20 000 172
5	28.75	31.43	35.27	40.66
10	14.38	15.71	17.64	20.33
20	7.19	7.86	8.82	10.16

increase as the number of unknowns is increased if sr remains constant. Choosing fewer segments per wavelength provides a coarser target representation, which will decrease accuracy. On the other hand, using too many segments per wavelength will result in too many unknowns.

Table 8.3 shows the frequencies for different numbers of unknowns and the corresponding number of edges per wavelength. The diameters of these spheres are two meters. At 15 GHz the diameter is 100λ, and at 30 GHz it is 200λ. In relationship to this table, a unit sphere with 20 million edges run at 30 GHz would have an average edge length between $\lambda/5$ and $\lambda/10$.

Once the appropriate frequency is chosen, the number of levels can be determined based on the size of the target. If we assume that the minimum box size at the finest level should be larger than 1.5 times the average edge length, then the maximum level should be

$$\mathbf{max_{level}} < \frac{\log \frac{D}{3L_e}}{\log 2} + 1 \tag{8.7}$$

where D is the side length of the smallest cube enclosing the target and L_e is the average edge length. Another practice is to keep the smallest box size above 0.15λ. It is not clear how small the finest level boxes can be, but several colleagues have suggested 0.25λ with the comment not to push it below 0.18λ for good accuracy. Binding the size to the average edge length is probably the best approach. Table 8.4 shows the four large sphere problems ranging from $100\lambda - 200\lambda$ with the number of levels used.

The next step is to estimate the memory based on a set of run parameters.

Table 8.4 Average edge lengths and lowest box sizes. Average edge lengths L_e and smallest box sizes are given in terms of wavelengths along with the maximum target size D. The estimated level is found using Equation (8.7). Note that the smallest box size is pushing the limits for accuracy for the 15 and 20 million sphere targets.

				Levels		Smallest
Edges	s	L_e	D	Est.	Used	Box
10 002 828	9.58	0.104	100	9.32	9	0.20
11 952 048	10.48	0.0954	100	9.45	9	0.20
15 052 800	7.84	0.128	150	9.61	10	0.15
20 000 172	6.78	0.148	200	9.81	10	0.20

For the 10-million-unknown problem (10M), 3 shared levels were used, and for all the others, 4 shared levels were used. The estimated memory for each run is given in Table 8.5 and the actual memory is shown in Table 8.6. Some explanation is needed to understand the memory trends.

Looking at the actual memory used (see Table 8.6), the total memory increased as the number of unknowns increased. However, the individual increases are highly correlated to the different run parameters along with the increase in the number of unknowns. For example, the memory for the O2I translators decreased between the 10M and the 12M spheres because an additional shared level was used. Since the 15M and 20M spheres each had 10-level trees, the number of filled boxes was about the same. However, because the frequency and accuracy digits changed in opposite directions, the number of samples needed at each level changed and the memory stayed about the same. If the digits of accuracy remained a constant 4.0, the 20.35 Gbytes would have increased to 33.14 Gbytes. It seems strange that the memory for the radiation patterns dropped sharply from the 15M sphere to the 20M sphere. However, the number of samples required at the lowest level dropped due to the relaxed accuracy for the 20M sphere. The near interaction memory drop between the smaller targets and the larger ones is due to the addition of a tenth level. This is a typical behavior where using one less level results in a factor of four more memory required for the near interactions. The block diagonal preconditioner memory follows a similar trend to the near interactions due also to the increase in

Table 8.5 Memory estimates for big sphere targets. Some of the key run parameters are shown.

Processes	16	24	33	32
Digits Accuracy	1.9	3.0	4.0	2.0
Shared Levels	3	4	4	4
Unknowns	10M	12M	15M	20M
Memory Type	in Gbytes			
Regular O2I	1.18	0.87	2.23	3.28
Box Patterns	6.16	6.29	21.56	20.35
Radiation Patterns	8.05	9.62	12.11	10.73
Near interaction	8.84	12.62	5.45	9.62
Block Preconditioner	0.69	1.03	0.50	0.78
GMRES Work Arrays	1.12	1.34	1.68	2.24
Geometry	0.56	0.67	0.84	1.12
Total	26.59	32.42	44.37	48.11

Table 8.6 Actual memory for big sphere targets.

Unknowns	10M	12M	15M	20M
Memory Type	in Gbytes			
Regular O2I	1.18	0.87	2.23	3.28
Box Patterns	6.16	6.29	21.56	20.35
Radiation Patterns	9.54	11.40	14.36	10.73
Near interaction	9.57	13.67	5.43	9.57
Block Preconditioner	0.90	1.27	0.56	0.96
GMRES Work Arrays	1.12	1.34	1.68	2.24
Geometry	0.56	0.67	0.84	1.12
Total	29.02	35.49	45.60	48.24

Table 8.7 Run times for big sphere targets.

Unknowns	10M	12M	15M	20M
O2I	348	191	465	983
Finest level	3189	2836	1513	1976
Core initialization	496	303	805	1334
CPU time in FMA	8798	7497	26 090	32 380
Processors	16	19	19	19
Processes	16	24	33	32
Iterations	50	51	62	64

number of levels used. The memory for the GMRES work arrays and the geometry follows the usual trend as expected.

After the memory was estimated and run parameters decided, the large spheres were calculated. These ultra-large-scale problems were run on the new SUN Blade 2000 cluster at CCEM. Table 8.7 shows the run times for each of the targets. All of the solutions were iterated to a residual error of 10^{-3}. As expected, the larger problems required more iterations. The outgoing to incoming (O2I) translator times seemed to follow the memory amounts. The finest level computation times are the average CPU times for each process and are for calculating the near interactions. The core initialization times are also average times per process. The reason that more processes were used than processors was to exploit the heterogeneous memory configuration of the SUN cluster and to keep each process below the 2 GB limit set by the MPI compiler. As expected, the CPU time in the fast multipole algorithm, where the iterations of the matrix-vector products are done, represents the lion share of the processing time. Also, since the network is not a dedicated supercomputer network, the communication times will vary on each run so these numbers are simply data points to provide an idea of the number of CPU seconds in different phases of the computation.

The Mie series can be used to calculate the scattering from a sphere for comparisons. In Figure 8.3, the data for each of these four large scale sphere problems are plotted against the Mie series. Table 8.8 shows the root mean square (RMS) error of the bistatic RCS computed using LSSP and using the Mie series.

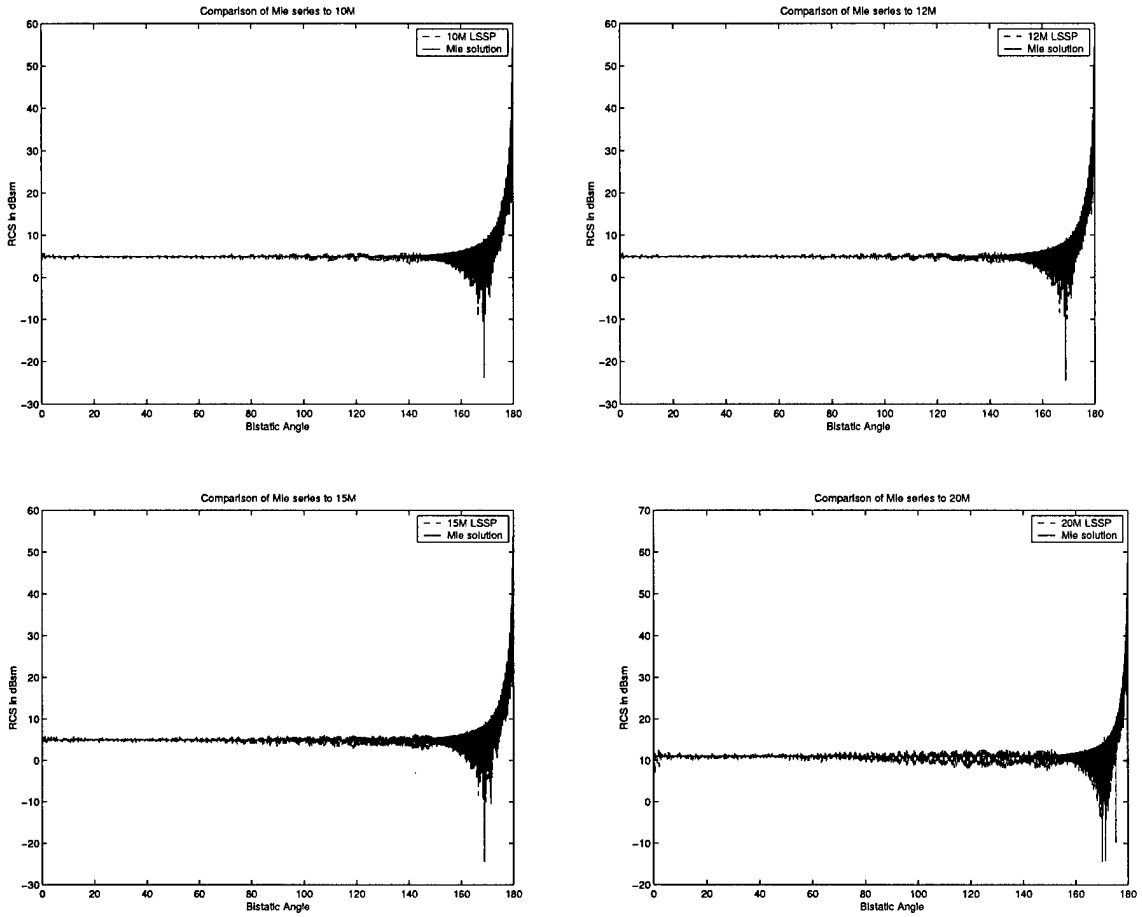

Figure 8.3 Bistatic scattering of large spheres.

Table 8.8 Root mean square (RMS) error for bistatic sphere data. This data is based on runs that were accomplished prior to the annihilation of the noise error bug described in Chapter 6.

Unknowns	10M	12M	15M	20M
RMS error (in dB)	0.498	0.467	0.688	1.086

Table 8.9 RMS error for bistatic sphere data – postbug fix. This data is based on runs that were accomplished after the annihilation of the noise error bug described in Chapter 6.

Unknowns	15M		20M	
	0°-90°	0°-180°	0°-90°	0°-180°
RMS error (in dB)	0.132	0.537	0.226	0.701

The 20 million sphere had twice the physical radius of the other three spheres so that is why the backscatter term is 10.99 dBsm ($10 \log \pi r^2$) instead of the normal 4.97 dBsm for a 1-meter-radius sphere. The excessive noise that increases with larger problems was discussed in Chapter 6. This data and the comparison to the Mie series is included to document the transition of large scale sphere problems with the noise error to the same problems after fixing the error.

After the noise bug was eliminated, the 15 million and the 20 million spheres were recalculated. Table 8.9 shows the RMS error of the data compared to the Mie series. The 20 million sphere took 62 iterations to converge, which is two less than when the bug was present. Likewise, the 15 million sphere took less iterations with only 56. For some reason, the bug correction seems to affect the number of iterations in a positive manner. All of the other details of these runs were essentially the same except that the 15 million unknown sphere was run with just 28 processes this time. Figure 8.4 shows the newly computed data compared to the Mie series.

8.4 Complex Targets

This section briefly highlights the extension of this work to large-scale complex targets. Spheres are easy to model and possess full symmetry and therefore do not qualify as complex targets. Cylinders are slightly more complex and several were run as part of the scaling studies discussed in Chapter 7. The missile-shaped target called pencil was calculated across a broad frequency range from 500 MHz to 12 GHz with the number of unknowns exceeding one million at the top frequency. Because larger-scale problems have been run, problems of this size seem fairly easy to run. The C-29 aircraft target was run using EFIE, since it is an open

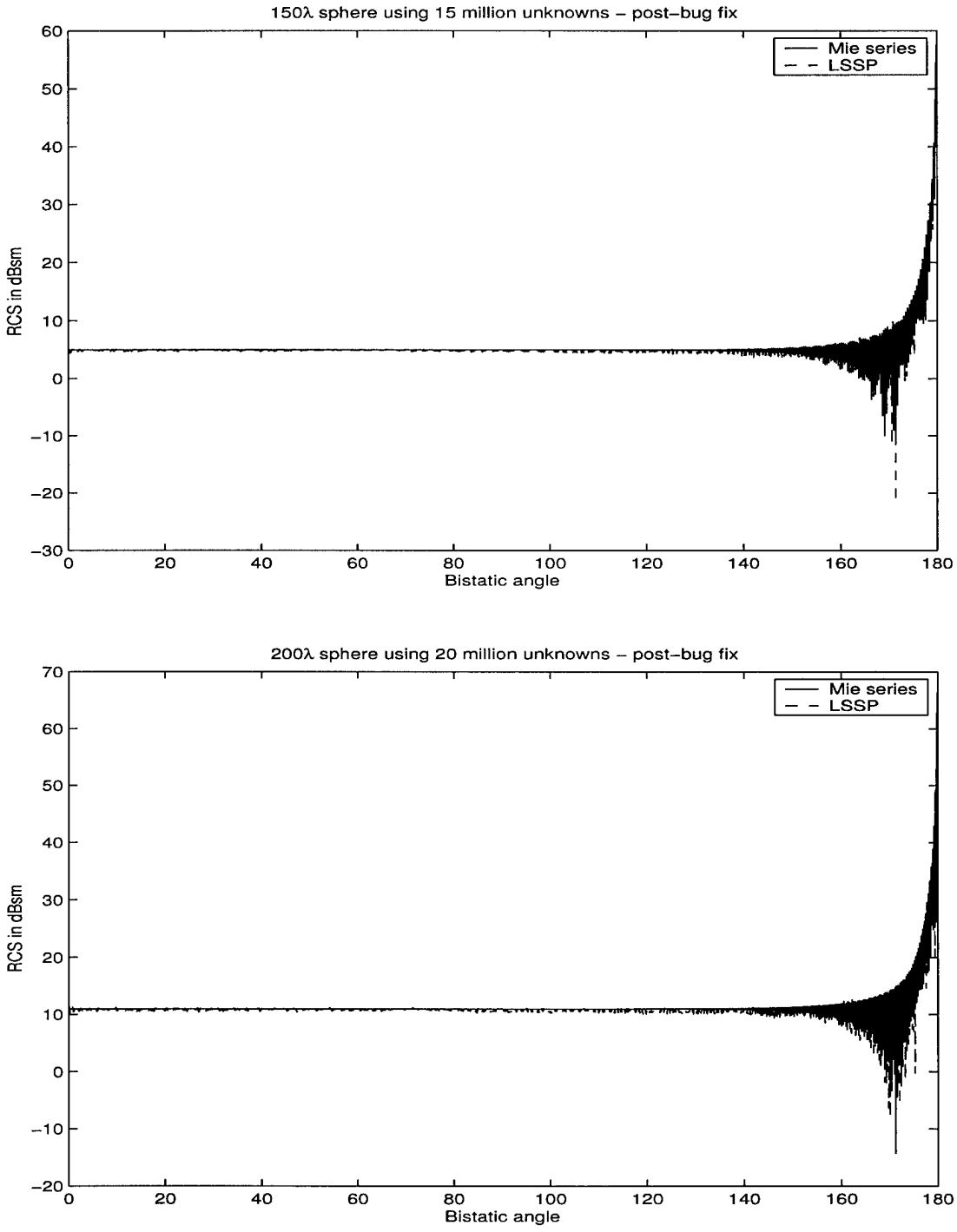

Figure 8.4 Bistatic scattering of ultra-large-scale spheres. The data for these plots was computed after the noise bug was fixed. The RCS of the LSSP data is compared to the Mie series.

target, but it did not converge after hundreds of iterations. The VFY-218 was run with 10 million unknowns on the CCEM SUN Blade cluster but had to be rotated geometrically to minimize the memory requirements. The data for this and a description of this technique is shown in Chapter 9.

CHAPTER 9

MEMORY REDUCTION THROUGH TARGET ROTATION

9.1 Introduction

Using a parallel implementation of the multilevel fast multipole algorithm (MLFMA) called LSSP, large-scale scattering problems can be solved [4]. However, as the problem sizes in terms of number of unknowns are increased, the memory requirements increase as $O(N \log N)$. In November 2002, a 200λ sphere with 20 million unknowns was solved using LSSP on a SUN Blade cluster with 10 dual-processor nodes. Careful memory optimization had to be considered since it required about 48 Gbytes of RAM. For realistic targets of a large-scale nature, like aircraft, the bounding box compared to the number of unknowns is much larger than that of a sphere. In many cases, we have found that the memory can be significantly reduced for realistic targets through geometry rotation.

This chapter illustrates how the memory required by some practical problems can potentially be cut nearly in half. The reduction in memory also leads to significant speed improvements for setup and initialization times including the computation for the finest-level near interactions. The number of iterations and the time in the fast multiple algorithm stay fairly constant with this new technique. The memory reduction can be realized by constructing the bounding box around the target such that the overall dimension of the cube enclosing the target is minimized. In the case of a sphere, the rotational symmetry gives rise to a single box size; however, with a missile-shaped target or aircraft, the box size containing the target is reduced when the target extends from corner to corner instead of side to side. This can be realized by a simple coordinate transformation or rotation of the target geometry.

9.2 Target Rotation and Bounding Box

Generally, a missile or aircraft target geometry will be constructed to point along the $\hat{\mathbf{x}}$-axis. Thus, the bounding box orthogonal to the cardinal directions will be a cube with a side length usually given by the length from nose to tail. Considering the shape of a missile, which we will model as a cone-capped cylinder, the bounding box is dominated by the missile length. If we consider an infinitely thin missile or line of unit length, such a target rotated 45° could fit in a cube of side length $1/\sqrt{2}$. With a rotation in θ and ϕ, it could fit in a box with a side length of $1/\sqrt{3}$. What this means is that the box size can potentially be reduced from unity to 0.707 or even 0.577 in the limit. Of course, realistic targets will require slightly larger boxes due to thickness and appendages such as wings or fins. However, the bounding box can still be reduced, which translates to smaller box sizes that require fewer samples for radiation patterns.

In Figure 9.1, the simple azimuthal rotation of 45° results in significant bounding-box size reduction for a cone-cylinder and the VFY-218 target. These targets represent a missile body and a fighter-type aircraft. The advantage of a single rotation in azimuth is that the aspect angles for the simulation are just an offset bias of 45°. If a rotation is done in both ϕ and θ, care must be taken in order to simulate the correct angles and polarization.

Table 9.1 shows the comparison of bounding box sizes for a line source, the pencil target, and the VFY-218. The line source represents the theoretical maximum bounding-box shrinkage. The width and height of length-dominant targets prevent maximum theoretical bounding-box reduction. The asymmetrical height of the aircraft target due to the vertical stabilizer leads to a larger θ-rotation to achieve the minimum bounding box. With a θ-rotation of 43°, the bounding box can be reduced to 10.4 m or 67% of the original size. The process of rotating the geometry into the smallest box could be automated.

9.3 Results

Once the target is rotated, it is possible to realize significant memory savings. The savings can only occur if the same number of levels is used in the MLFMA tree. If the finest level boxes are too small in relationship to the average edge length, one level fewer will be used and the near-interaction memory will increase

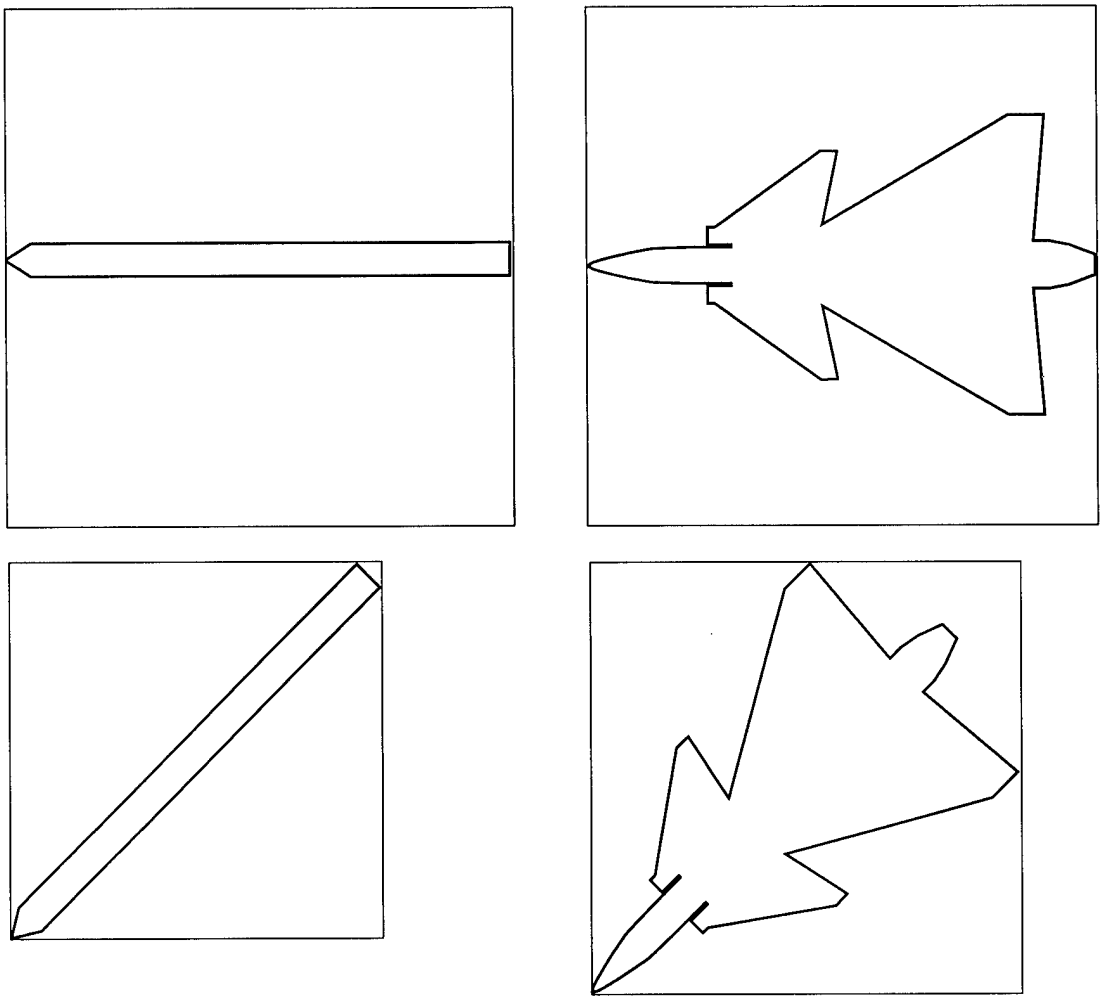

Figure 9.1 Geometry rotation of 45° in azimuth shrinks the bounding box. The box shrinkage is 73% for the cone-cylinder (pencil target) and 84% for the VFY-218.

by a factor of four. Therefore, an initial calculation should be made to determine if bounding box reduction can be used with the same number of levels as before the target rotation. In practice, we use the rule-of-thumb that the smallest box should be larger than 1.5 times the average edge length. The smallest box size and its separation from interaction boxes determines the error limits [3]. If using the same number of levels would make the finest level box sizes too small, the minimum box size could be used at the finest level. LSSP has the option to create the tree using a smallest box size or to use the maximum bounding-box dimension with recursive subdivisions for a specified number of MLFMA levels. If the smallest box size is

Table 9.1 The bounding box shrinkage through geometry rotation. This shrinkage for realistic targets can be significant. The final row illustrates a rotation in elevation followed by a rotation in azimuth. This produces the box diagonal for the line source and equivalent lengths along the cardinal directions.

Rotation	Line Source		Pencil		VFY-218	
0	1.00	100%	3.17	100%	15.5	100%
$\phi = 45$	0.707	71%	2.31	73%	13.0	84%
$\theta = 35$, $\phi = 45$	0.577	58%	1.91	60%	11.2	72%

specified, the top-level box will be larger than the rotated target's bounding box.

Applying the target rotation to the pencil and VFY-218, the reduction in memory is given in Table 9.2. In the case of the pencil target, the box shrinkage to 71% associated with a single rotation of 45° resulted in 60% of the memory requirement before rotation. This is further reduced to 47% when the pencil extends from opposite corners of the box after a rotation in θ and ϕ. For the VFY-218, the first rotation as illustrated in Figure 9.1 uses only 80% of the original memory while the minimum box size results in 58% of the original memory requirement.

Table 9.2 Memory reduction for pencil and VFY-218. With a simple azimuthal rotation, the memory is reduced by 40% for the pencil target and 20% for the VFY-218.

	Pencil	VFY-218
Unknowns	291 774	3 114 762
Levels	8	9
Original Memory	1311 MB	11.57 GB
$\phi = 45$ Rotation Memory	787 MB	9.25 GB
Minimum Bounding-Box Memory	610 MB	6.67 GB
Finest Level Minimum Box Size	0.15	0.20

Figure 9.2 shows the comparison of RCS data produced using a rotated VFY-218 and the original target. A residual error of 0.003 was achieved in 140 iterations. The mean square error is shown in the caption, and it is due to the residual and to smaller box sizes (fewer samples).

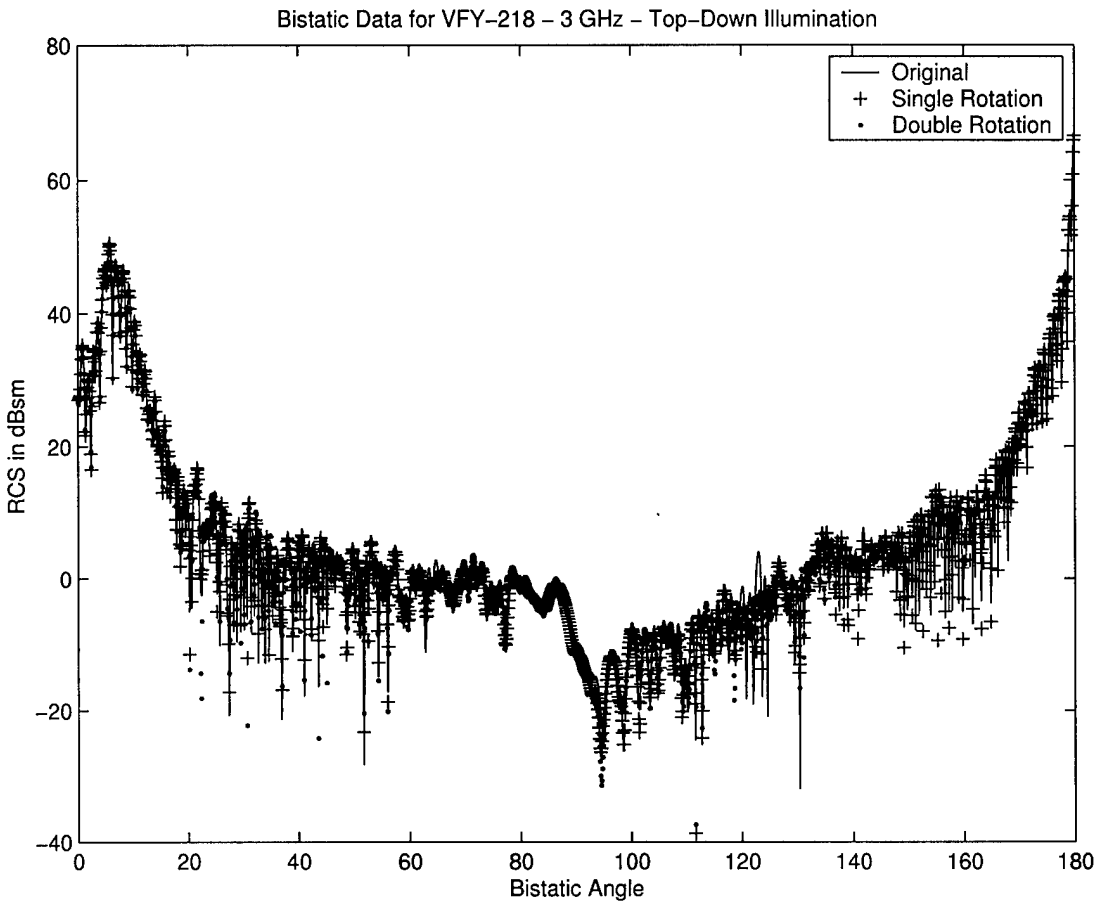

Figure 9.2 Bistatic RCS data produced using original and rotated targets. The root mean square error between the original and the two rotated versions is 2 dB and 2.4 dB.

This technique was used to simulate data for a 10-million-unknown VFY-218 on the CCEM SUN Blade cluster using significantly less memory and the same number of digits accuracy as previous runs done using FISC on a SGI Origin 2000 shared memory supercomputer [5]. The memory requirments were reduced from 46 GB to 36 GB. The data is shown in Figure 9.3.

9.4 Conclusions

This chapter has introduced a new technique to reduce the memory required by MLFMA. Reductions on the order of 50% are possible for targets with a dominant length whose bounding boxes can be shrunk significantly [23]. The memory is

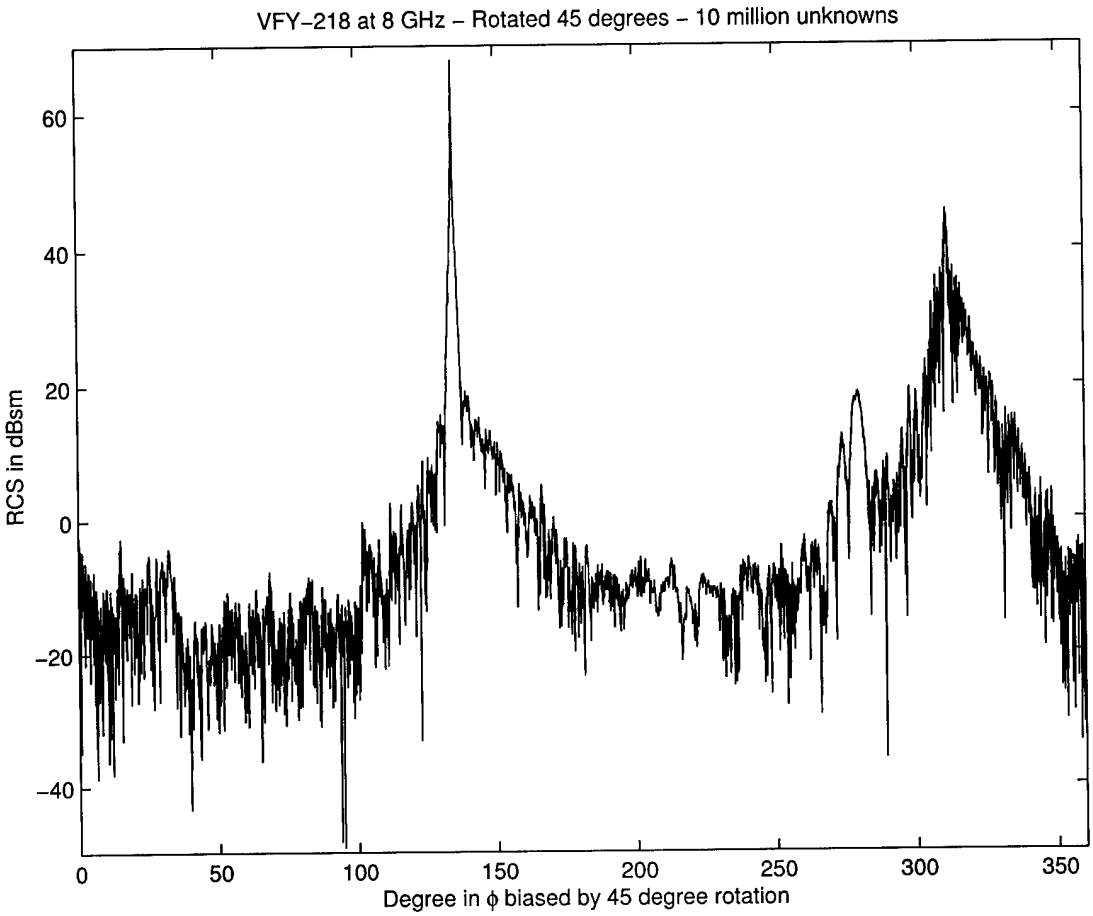

Figure 9.3 Bistatic RCS data produced on rotated VFY-218 with 10 million unknowns.

dramatically reduced for the radiation patterns and near neighbor interactions. The radiation pattern memory is decreased because fewer samples are required at the lowest level. Near-neighbor interactions decrease because the number of near-neighbor, method of moment elements decreases. The savings is achieved when the number of levels in the MLFMA tree remains constant with bounding box reduction. Target rotation in azimuth allows for simple application while geometry rotations in both azimuth and elevation require more complicated treatment of aspect angles and polarization. This technique to reduce the memory will allow for larger realistic problems to be solved in the near future.

By fixing the lowest level box size and increasing the effective bounding box, while adding an additional level without rotating the target, leads to a reduction

in the memory. However, this reduction is not as large as can be produced by rotating the geometry. The number of samples required at the lowest level is the same as the rotated target and lower than the normal running configuration. The reason this memory is still higher than the rotated target is attributed to the extra level. At the uppermost level, the translators are significantly larger and must be stored in the memory.

CHAPTER 10

TARGET-FREQUENCY-BASED ALGORITHM SELECTION

10.1 Introduction

Many people that use computational electromagnetics to produce data are interested in two factors: speed and accuracy. Likewise, individuals that use simulated data are interested in the same factors. To appreciate the tradeoff, it is imperative that the parameters used to create the data be understood. Often these parameters are used to control these factors. Each computational method has approximations and these are the error sources (see Chapter 4). The purpose of this chapter is to compare data produced using high frequency techniques and the multilevel fast multipole algorithm using the codes described in Chapter 5. High frequency techniques break down as the target becomes smaller in terms of wavelengths. The method of moments stretches the computing resources in real-world problems. A missile-shaped target called the pencil is used as a test object to compare data produced using Xpatch, FISC, and LSSP.

10.2 Speed Versus Accuracy

The rule of thumb, 'you get what you pay for,' applies to data produced through computer simulation. Unfortunately, the last few decibels of accuracy have a disproportionately high cost. Often times when codes are benchmarked and compared, this fact is overshadowed by the lightning speeds of fast, but approximate, algorithms. Fast algorithms are not exempt from the high costs of accuracy. Some algorithms cannot control the errors to a finite precision. This fact applies, in particular, when codes are used at or beyond their limits.

On the topic of limits, often times, the limits of a particular technique are not clear. For example, in the early days of Xpatch development, I remember hearing claims that data could be produced on aerial targets down to 500 MHz. Later, some qualifying statements were made that missiles might only be predicted down

to S-band (2-4 GHz) for good accuracy. It is obvious that two distinct issues are playing out. First, people always want to stretch the limits of a code. This natural tendency often leads to unreliable data. Sometimes the data compares favorably to data produced by a trusted source and is deemed to produce accurate data within a new set of limits. Another target, where valid data is not available, may be run in the same way and the data trusted based on this previous evidence. When one climbs out on a limb and the limb does not break, does this imply that one's friend can do likewise? Clearly this is not the case. Eventually someone will get hurt. The impact of erroneous data can be severe in terms of loss of life or resources.

With so much at stake, one might wonder why accuracy is not the driving factor in computational electromagnetics. The potential impact of erroneous data might be outweighed by the requirement for timely data. Nearly a decade ago, I was working on a project and got a call from the Pentagon late on a Friday afternoon. A request was made for the signature level of a particular target. It became readily apparent that the requester knew little about signature data; however, he was briefing a general in 10 minutes and needed data immediately. When I tried to explain that signature data is not just a single number, he asked me if I wanted him to guess or if we could make a better assessment. We were in a better position to make a qualified assessment. If a pilot is flying over enemy territory and needs to know when he will be detected, timely and accurate data are essential. However, if the data is not timely, mission planning is problematic. It may be better to be close and on time than accurate after the data was needed.

In considering speed and accuracy related to simulated data, it is important to understand how the data will be used. The end use of the data is essential before deciding on the run parameters. For example, in Chapter 8, data for a 20-million-unknown sphere produced using MLFMA is shown. Also, previously published work demonstrating the capability to do an aircraft target with 10 million unknowns at X-band using first-order physics has been shown [5]. These kind of runs push the envelope and are not done with the maximum accuracy. However, they demonstrate that the technology can be pushed forward. In the case of MLFMA, the accuracy can be controlled with sufficient computational resources and time.

In the remainder of this chapter, data on a single target will be shown and analyzed. This is one target, and the conclusions about the capabilities of any

Table 10.1 Size of pencil target in wavelengths at the tested frequencies.

Frequency	Length	Diameter
500 MHz	5.3	0.3
1 GHz	10.6	0.7
2 GHz	21.2	1.3
3 GHz	31.7	2.0
6 GHz	63.5	4.0
12 GHz	126.9	8.0

particular code cannot be accurately determined using one test object. A large set of unique test objects should be studied in the same manner as approached by this research in order to make solid conclusions on the limitations of a particular method. Since the focus of this research is to help qualify when MLFMA should be used, the data will be considered with regard to speed and accuracy.

10.3 Results

Data was collected on the pencil target shown in Figure 7.7 across a broad frequency regime. The data produced using MLFMA was produced with high accuracy settings and is considered to be accurate. The data simulated using Xpatch is also considered accurate in that it was run with standard settings. However, it was run intentionally outside the limitations of high frequency techniques. Table 10.1 shows the length of the missile-like object in terms of wavelengths at the frequencies at which it was tested. The pencil target is formed with a 3-m capped cylinder with a 0.1-m radius and a tip extending 0.173 m pointing towards 0° azimuth and elevation. The conical tip is designed to produce a specular backscatter at a 60° angle.

The broadside is sufficiently long to produce accurate high frequency data above 1 GHz while the tail should be accurate above 12 GHz. The data was run using both vertical and horizontal polarizations. A complete set of data can be found in Appendix A and includes monostatic patterns as well as bistatic data. The bistatic data was calculated with incident angles every 30° in azimuth from 0° to 180° including a nose-on grazing azimuth angle of 3°. Several samples are shown

here and discussed in detail.

Examining the monostatic data first, the broadside peak is easily predicted across the range of frequencies. This is not surprising since the dominant scattering mechanism is specular in nature. The tip is not accurately predicted across the frequency range and suffers the most error at lower frequencies. The tail is essentially a circular flat plate, which by 2 GHz becomes the dominant scattering in the tail region. To illustrate the accuracy of Xpatch across this broad frequency range, three plots are shown.

In the first plot, Figure 10.1, the horizontal polarization at 1 GHz is shown. The broadside peak is captured accurately by Xpatch including some sidelobes. The sidelobes towards the front of the target have good agreement between 50° and 90° but only two sidelobes away from broadside to the back of the pencil target are predicted well by Xpatch. The peaks seem to be under-predicted over a wide range of angles out to 140°.

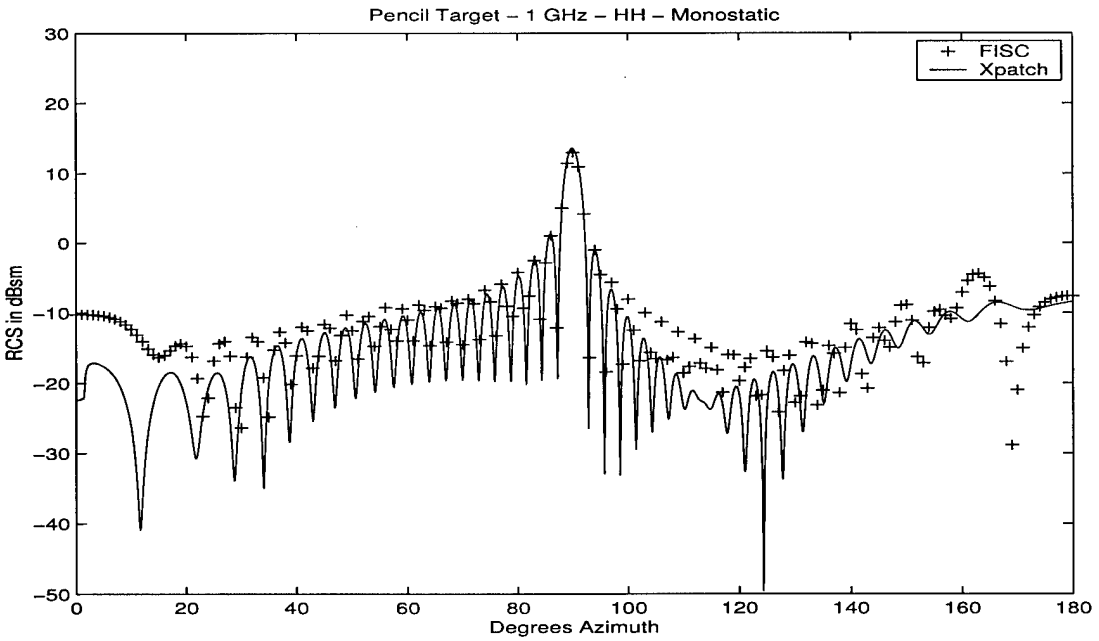

Figure 10.1 Monostatic scattering of pencil at 1 GHz – HH polarization.

The nose region, which is often most problematic, is significantly under-predicted by Xpatch, and the structure is quite different. The disparity in the nose region is more than 10 dB on average. Since Xpatch does not predict tip diffraction or

multiple diffractions, this region is quite difficult to accurately simulate. Xpatch does not correctly capture the physics of the problem at this frequency on the nose. The tail aspects are a similar story. Since the backscatter of the circular flat plate is not large enough to be the dominant scattering source, a lobe structure at grazing incidence is observed. Xpatch cannot predict the amplitude or position of the first peak of this scattering mechanism, which is associated with waves that travel along the long missile-like body and diffract off of the edge formed by the nose-cone.

The second example for illustration is shown in Figure 10.2. In this plot, the same HH polarization is shown as the previous plot at 1 GHz. The agreement of Xpatch with FISC is excellent across nearly the entire set of aspect angles. The only problem area is the nose inside of 15°, which is still underpredicted by Xpatch. The lobe structure is much finer which is indicative of a region where high frequency methods are appropriate. The FISC data was only produced every degree due to the cost of monostatic data requiring the iterative solution of multiple right-hand-sides corresponding to each aspect angle. Fortunately, FISC employs a method to use the solution of the previous angle to significantly cut down on the number of iterations required. The description of the method to provide the iterative solver with an initial guess is described in [5].

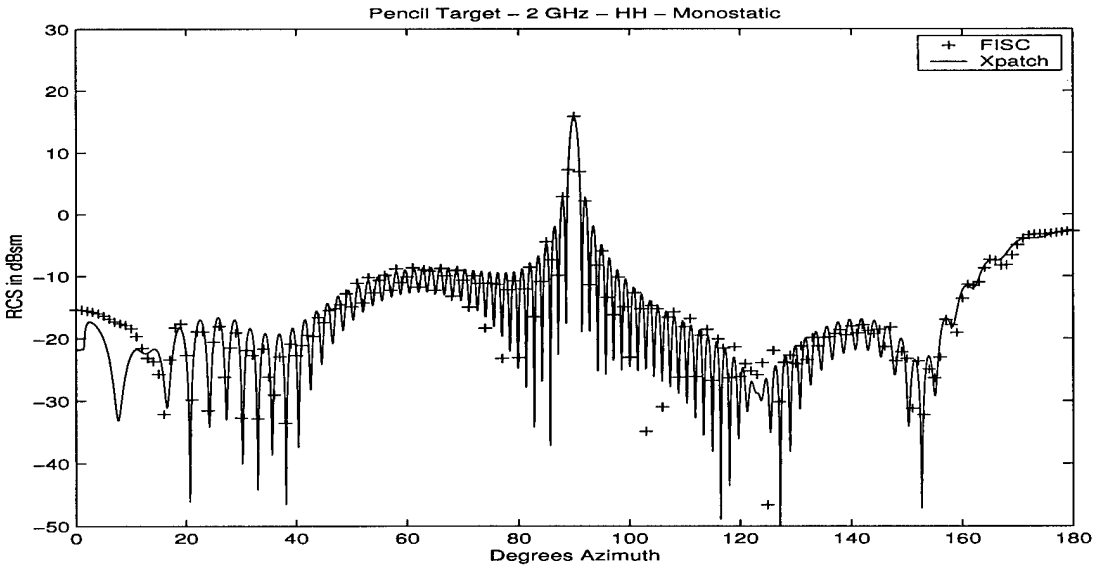

Figure 10.2 Monostatic scattering of pencil at 2 GHz – HH polarization.

Besides the accurate prediction of the broadside, the wide lobe associated with the specular scattering off of the nose-cone begins to emerge at the appropriate 60° angle. This lobe will increase in amplitude and become narrower as the frequency is increased. Distinct specular scattering typically is observed as the target size grows in terms of wavelengths. The length of the nose-cone is much smaller than the length of the cylindrical body, which explains the difference in width and height of these main specular lobes compared. Looking at the RCS at the back end, the dominant scattering of the circular flat plate is predicted very well by Xpatch.

The final example of monostatic data shown in this chapter is captured in Figure 10.3. FISC was not able to run this data on the SUN Blade cluster due to memory limitations and a single node requirement. FISC can only be run in parallel on a shared-memory machine and since this target involved more than a million unknowns, the memory exceeded its limitation on a single node. This data could have been run with FISC on a 64-bit shared memory machine. However, this was a good time to showcase the ability of LSSP to compute the data across the distributed machines. This data was fairly expensive in terms of processor time.

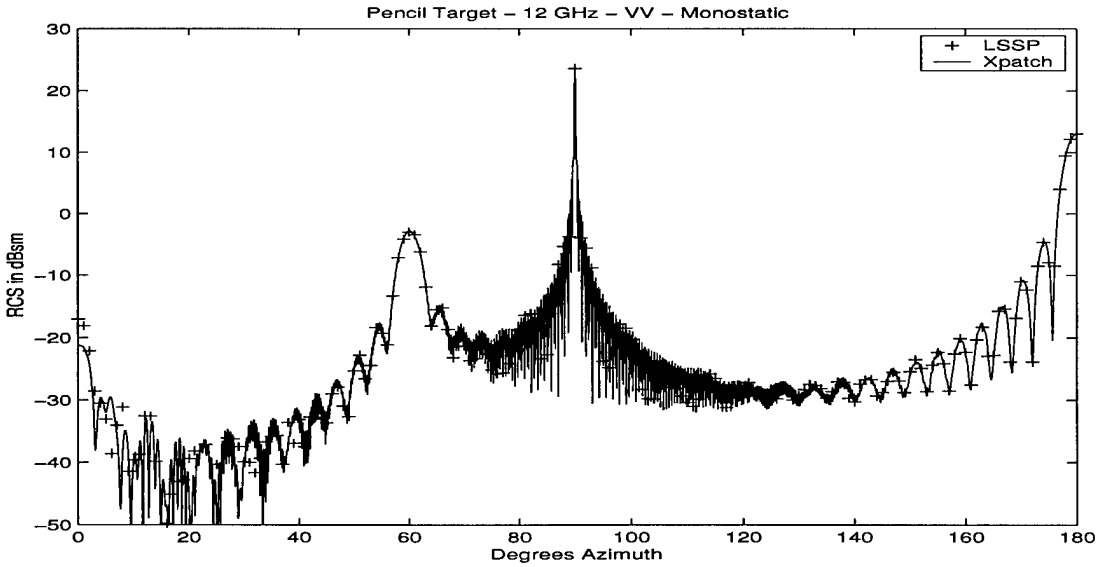

Figure 10.3 Monostatic scattering of pencil at 12 GHz – VV polarization.

Monostatic data for large-scale problems using MLFMA costs much more than Xpatch and only buys accuracy in the regions where specular scattering is not dominant. Also, with larger problems, significantly more points are necessary to

capture the fine lobe structure associated with the phase addition and cancellation of scattering mechanisms. In the case of LSSP, the previous angle solution is not used to seed the iterative solver so each aspect angle takes a full set of iterations. Fixing this in the future would be a significant improvement.

Figure 10.3 really illustrates the value of Xpatch in accurately predicting a majority of the scattering regions. With the exception of the nose, the entire set of data is excellent. The three dominant specular scattering regions are the nose cone which backscatters at 60°, the broadside at 90°, and the tail cap at 180°. All of these regions are based on specular scattering. The tail is a flat-plate scatterer and the broadside is a singly curved surface. The nose cone is a singly curved surface with a changing radius of curvature that decreases to zero at the tip. Common, hip-pocket formulas can be used to predict the amplitudes of each of these peaks [24].

More than 90% of the 181 monostatic points calculated using MLFMA were not necessary at this frequency. Computational resources would be more effectively used to predict the nose region with LSSP at finer increments and the rest of the target using Xpatch. This would be the most efficient way to acquire the most accurate data at the finest resolution with the lowest cost. The data from these multiple sources could then be merged together.

Monostatic data is very different from bistatic data. Xpatch does not seem to predict bistatic data as well as it can predict monostatic data. This is due, in large part, to the nonspecular nature of the scattering found in many regions of a bistatic data set. The single backscatter point in a bistatic data set will be the same as that produced in a monostatic run. However, it seems that if this point is off by a lot, the rest of the bistatic data will have a higher error level. When analyzing bistatic data, there are several types of scattering that are observed. The first, a major difference from monostatic data, is called the forward scatter region. This is the data in the opposite direction of the incident wave. It is generally a large and narrow scattering effect. The specular scattering shows up in bistatic angles that follow Snell's law, which states that the angle of incidence equals the angle of reflection. These features will be illustrated in forthcoming plots. Before showing the results of the bistatic comparison of high frequency methods to MLFMA, we note that FISC and LSSP can easily produce bistatic data since it involves only a single right-hand-side associated with the angle of incidence. The surface currents

are determined and the scattering in any direction can be calculated. The process in Xpatch requires careful tracking of the shooting and bouncing rays to calculate the scattering in a certain direction.

The first bistatic comparison is made at broadside incidence. Figure 10.4 shows some surprising results at 12 GHz where the data is expected to match closely. The unexpected difference can be seen in the forward scatter region located around 270°. Xpatch does well on the side being illuminated except a small region on the nose, but the opposite side of the pencil in the forward scatter region has significant errors. The logical explanation is that Xpatch is not capturing the physics associated with the forward scatter region at broadside illumination.

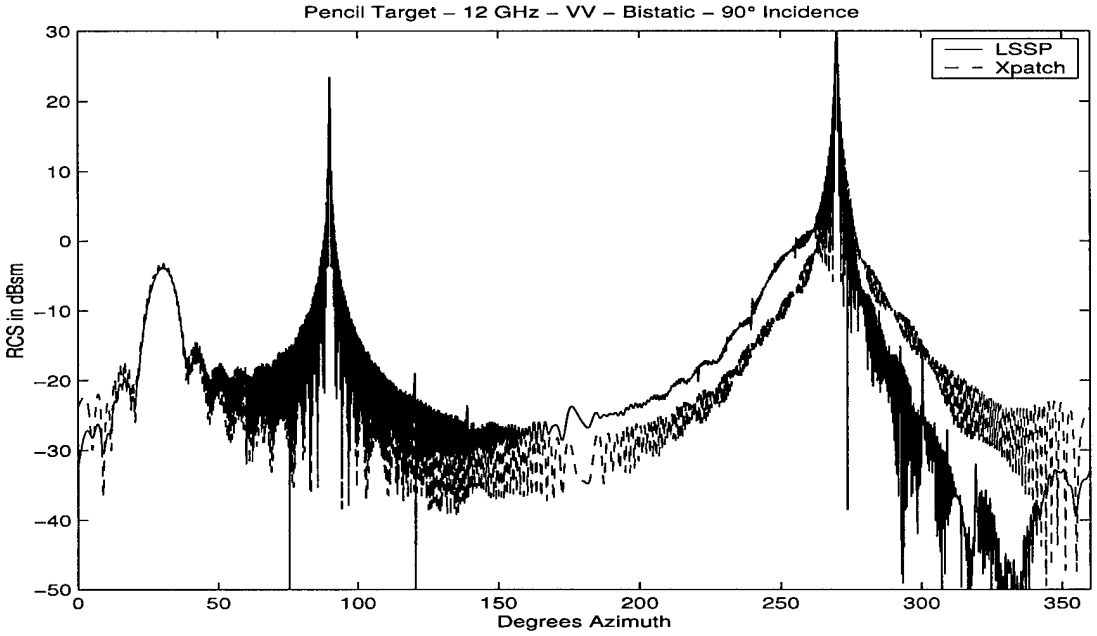

Figure 10.4 Bistatic scattering of pencil at 12 GHz – VV polarization – 90° incidence.

Related to the 2-GHz data previously shown in Figure 10.2, the bistatic data for 180° incidence is shown in Figure 10.5. Two things can be noted with respect to this graph. First, the plots are symmetric, which is expected. The other observation we make is that the fine structure and morphology of the Xpatch data differs significantly in the forward scatter hemisphere.

As a final analysis to some of the bistatic data, it is interesting to observe

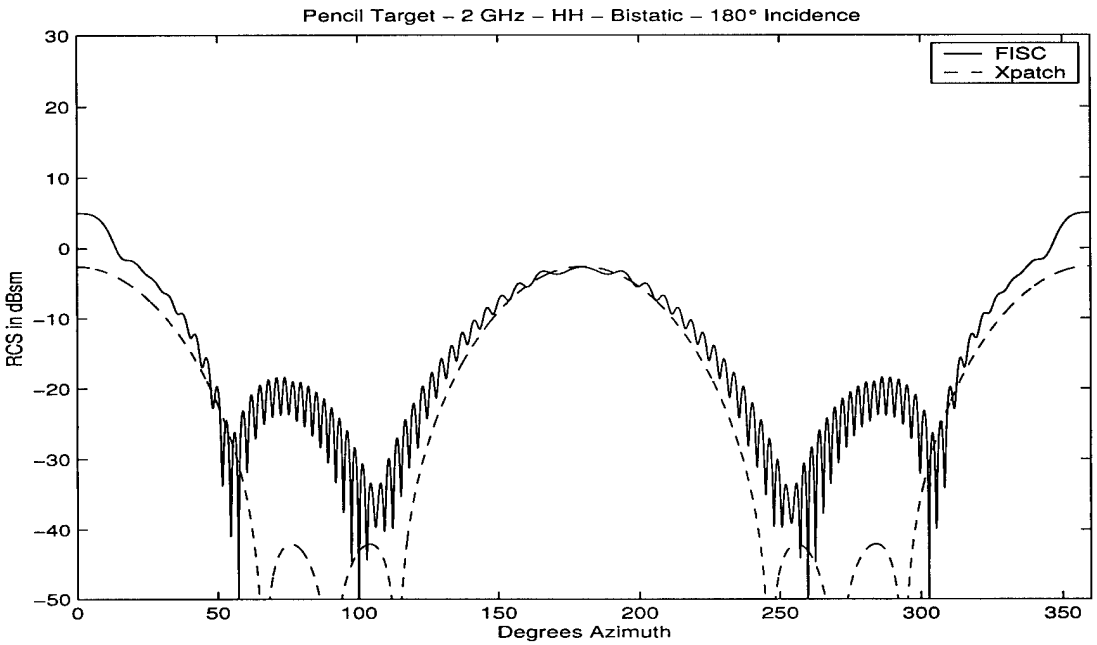

Figure 10.5 Bistatic scattering of pencil at 2 GHz – HH polarization – 180° incidence.

the scattering mechanisms as a function of bistatic angles. To illustrate this, Figure 10.6 shows data at 6 GHz for the incident angles of 0° to 150° every 30° for the VV polarization. A lot of these data agree well between Xpatch and FISC.

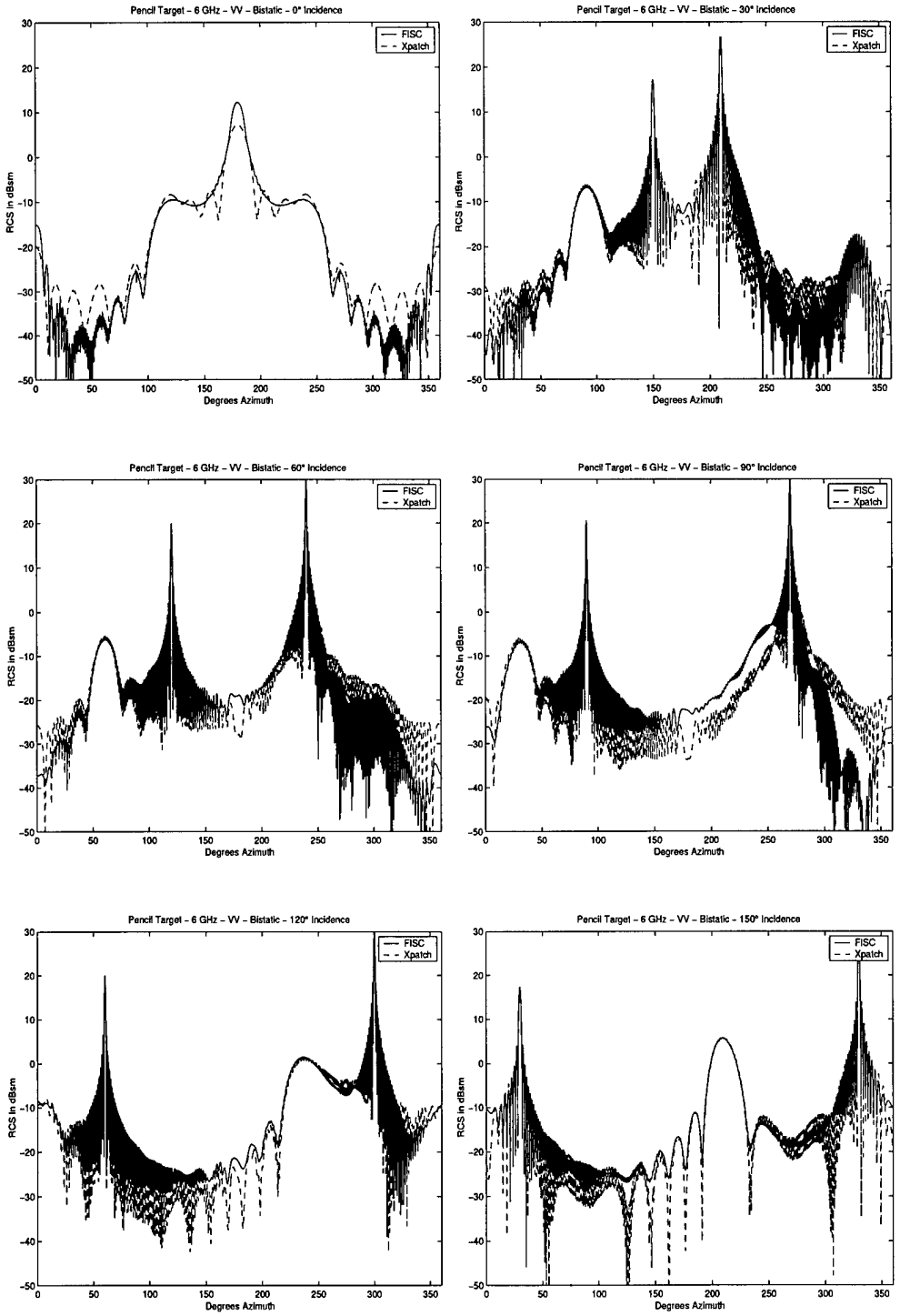

Figure 10.6 Bistatic scattering of pencil at 6 GHz – VV polarization – multiple incident angles.

Table 10.2 Specular scattering directions for the pencil target. These directions are based on the surface normals of the target and the incident wave directions.

	Pencil Normals		
Incident Angle	60°	90°	180°
0°	120°	180°	NA
30°	90°	150°	NA
60°	60°	120°	NA
90°	30°	90°	270°
120°	300°	60°	240°
150°	NA	30°	210°

In the 0° elevation plane, there are five normals to consider for specular scattering. The nose cone accounts for +/− 60° and the cylindrical body accounts for +/− 90° while the back cap is at 180°. According to Snell's law, the position of the specular scattering from each of these surfaces can be calculated and is found in Table 10.2. Only the illuminated surfaces with normals between 0° and 180° are shown in the table.

The position of each specular peak can be seen in Figure 10.5 and since the frequency is the same, the basic nature of the peak remains fairly constant as far as amplitude and width are concerned. The forward scatter is the dominant peak in each plot and shifts to the right by 30° with each step in incidence of 30°. The tip and tail regions seem to be the most sensitive to differences between Xpatch and FISC.

10.4 Conclusions

Through the process of qualitatively comparing the Xpatch data to the FISC or LSSP data it is clear that certain aspect angles are accurately predicted using high frequency methods. High frequency techniques are ultrafast compared to method of moment techniques even when accelerated using MLFMA. This particular target highlighted the fact that specular scattering is easily predicted using high frequency techniques, whereas grazing angles and tips are problematic, and in cases when specular scattering is not the dominant mechanism, full-wave solvers should be

used for increased accuracy. With the exception of the tip region out to about 10°, the monostatic data at 2 GHz and higher was quite reasonable. Data below this frequency regime for a target of this size and shape is only suitable for the specular scattering inherent at the broadside.

The bistatic data was more unpredictable using Xpatch. It appeared that when the incident direction correlated with accuracy in the monostatic data, the bistatic data tracked the baseline FISC or LSSP data. Bistatic specular regions were predicted well by Xpatch, but forward scattering regions often had differences in the peak amplitudes. Interesting scattering mechanisms were observed as the bistatic angle was varied. Agreement was best at the higher frequencies in the range. The Xpatch data seems to have an anomaly in the horizontal polarization for the bistatic data when the illumination was at nose-on or tail-on. The data lacked the expected symmetry of this problem, which should be analyzed further. At low frequencies, with incident directions poorly predicted by Xpatch, the bistatic data lacked the finer signature structure.

Monostatic data is costly for algorithms based on the method of moments. Xpatch is superior for predicting specular scattering regions including the neighboring sidelobes. Outside of these regions, if accurate data is required, a full-wave solver incorporating the physics of the problem should be used. Based on the frequency and shape of the target, regions can be divided where high frequency techniques are reliable and fast. Specular scattering can be determined by studying the normals of a target and the scattering amplitudes. As the frequency increases, these spikes become taller and narrower. As the target increase in electrical size, the quality of the Xpatch data clearly increases. By using Xpatch to predict the trusted areas, these regions can be quickly calculated while the tougher regions can be predicted using rigorous methods that include small effects that may become dominant at certain aspects and frequencies.

CHAPTER 11

CONCLUSION

11.1 Accomplishments

This research has helped me slice to the center of this technology to a depth that I only dreamed of reaching. Understanding the fine details of MLFMA forms an essential foundation for improving the technology and pushing the state-of-the-art forward. Controlling the errors in MLFMA is important for high accuracy applications. Since this technology is based on MoM, we desire high accuracy solutions. Unlike MoM, we want the results fast.

This dissertation illustrates that the error in the translation operator can be controlled to a reasonable precision, but to do so may require the use of additional buffer boxes and less levels. Additional buffer boxes and fewer levels result in a tradeoff with speed. Speed versus accuracy is often the reason why careful parameter selection must be done. We have shown that the error control in 2-D problems applies to 3-D problems. The minimum error in the translation as a function of box sizes was shown. The minimum error forms a boundary between the controllable and uncontrollable regions. Understanding the error sources is critical to achieving high accuracy solutions.

One of the significant contributions of this work in improving the data produced using this algorithm was the annihilation of the bug that produced excessive noise in the data for large-scale problems. This required a great deal of persistence and hard work to find. In the process several other bugs were found and corrected. The key thing learned is the importance of maintaining consistency in the precision of constants that may be defined in individual subroutines that are blended together as in the case of ScaleME and TRIMOM. The impact of fourth digit inconsistencies is not noticed in small-scale problems but emerges as a skeleton in the closet with large-scale problems.

Since we want the simulations to be fast as well as accurate, we need to have a clear understanding of how to set the parameters to achieve the desired level of

accuracy in the best possible time. As the problem size increases, parallel methods must be employed. These methods come with their own specific challenges and limitations. This research has shown the ability to solve large-scale problems using the parallel algorithm called ScaleME. Many computer hours have been used to test this code.

We have demonstrated the ability to solve up to 20 million unknowns using this technology. Previously, the record was 10 million unknowns. As a part of this process, a method to predict the amount of memory required for a target with a set of parameters had to be developed. Using this predictive tool, targets can be carefully fit into memory while adjusting the parameter inputs to reach the proper balance. Scaling and parallel efficiency were studied using LSSP.

Finally, we have discovered ways to optimize computations and to reduce memory requirements. Through target rotations, the memory can be greatly reduced. Such target manipulation allowed for the solution of a 10 million unknown VFY-218 with 10 Gbytes less memory than was previously required for the same accuracy settings. Also, bridging the gap between high frequency methods and the method of moments led to the ideas of algorithm selection based on target geometry and frequency. Running MLFMA in regions where high frequency techniques are both accurate and fast makes little sense; using it in regions where errors in high frequency predictions are high is effective.

11.2 Future Work

With the constant inconsistency error eliminated, the new approach to error control could be implemented in LSSP. Since the error sources associated with the translation operator are now better understood, it should be possible to study the effect of nonuniform sampling at the lowest levels and explore the tradeoffs between accuracy and buffer boxes. Examining the impact of residual error on RCS calculations is another possible area to study.

A continued study of LSSP on additional geometries and different platforms should be pursued. Extensive testing will increase the robustness of the code. This testing will produce data on runtimes and parameters. Currently, a good way to estimate run times does not exist. Furthermore, the run times are highly dependent on the parameters and the target geometry. It would be useful to

develop an approximation of expected run times on different platforms.

There are several relatively straightforward projects that would be valuable in the development of LSSP. The input page needs work and should be modified to be more consistent with FISC. Part of this standardization means using azimuth and elevation as inputs rather than θ and ϕ. Also, an input file should contain the name of the geometry file rather than requiring the name of the geometry to be the same as the configuration file name. The initial guess in the iterative solver should be adjusted to take advantage of the previous monostatic solution as in FISC runs. This will increase the speed performance in the acquisition of monostatic data by reducing the number of required iterations. Also, the 10-level limitation should be addressed so that more than 10 MLFMA levels can be used for ultra-large-scale problems. Lastly, certain run checks should be made to verify that the geometry is appropriately discretized for the desired frequency. In summary, the features in FISC, where possible, should be incorporated into LSSP to make it more capable and standardized.

Finally, solving larger, more complex targets should be pursued. As part of such a quest, comparing high frequency calculations with MLFMA should reveal areas where MLFMA must be used for accuracy. It may be possible to design a methodology to determine the appropriate method to use for accuracy and efficiency. As one can see, there are many possible future research contributions. We plan to maximize the contributions to further the technology by continued deep study into large-scale problems using these new error control methods with the parallel implementation of MLFMA in the code LSSP. Of course, eventually the capability to do materials other than perfect electric conductors will require significant research and testing. The work accomplished as part of this dissertation positions future researchers to take LSSP to new levels of performance.

APPENDIX A

DATA ON PENCIL TARGET

This appendix contains data discussed in Chapter 10. Figures A.1 – A.22 represent the complete set of data collected to study the pencil target between 500 MHz and 12 GHz. The Xpatch data was produced using the target I provided to SAIC. Dr. Steve Kosanovich (of SAIC) generated the Xpatch data according to the matrix I provided and with the understanding that the test target and frequency range were designed to push beyond the limits of a high frequency code to produce valid data on this particular target.

Data was simulated at the following frequencies: 500 MHz, 1 GHz, 2 GHz, 3 GHz, 6 GHz, and 12 GHz. Bistatic data was produced with incident waves every 30° from 0° to 180° plus a grazing angle of 3°. In each case, both co-polarizations (VV and HH) were simulated. VV represents both vertically polarized transmitted and received waves. HH refers to the horizontal polarization. Monostatic data was produced at each frequency between 0° and 180° rather than a complete circle, as in the case of the bistatic data, due to symmetry. The bistatic data was collected in all cases every 0.1°. The monostatic Xpatch data was also collected at the same angular resolution; however, due to the expense of monostatic data using FISC or LSSP, this was relaxed to every degree in azimuth.

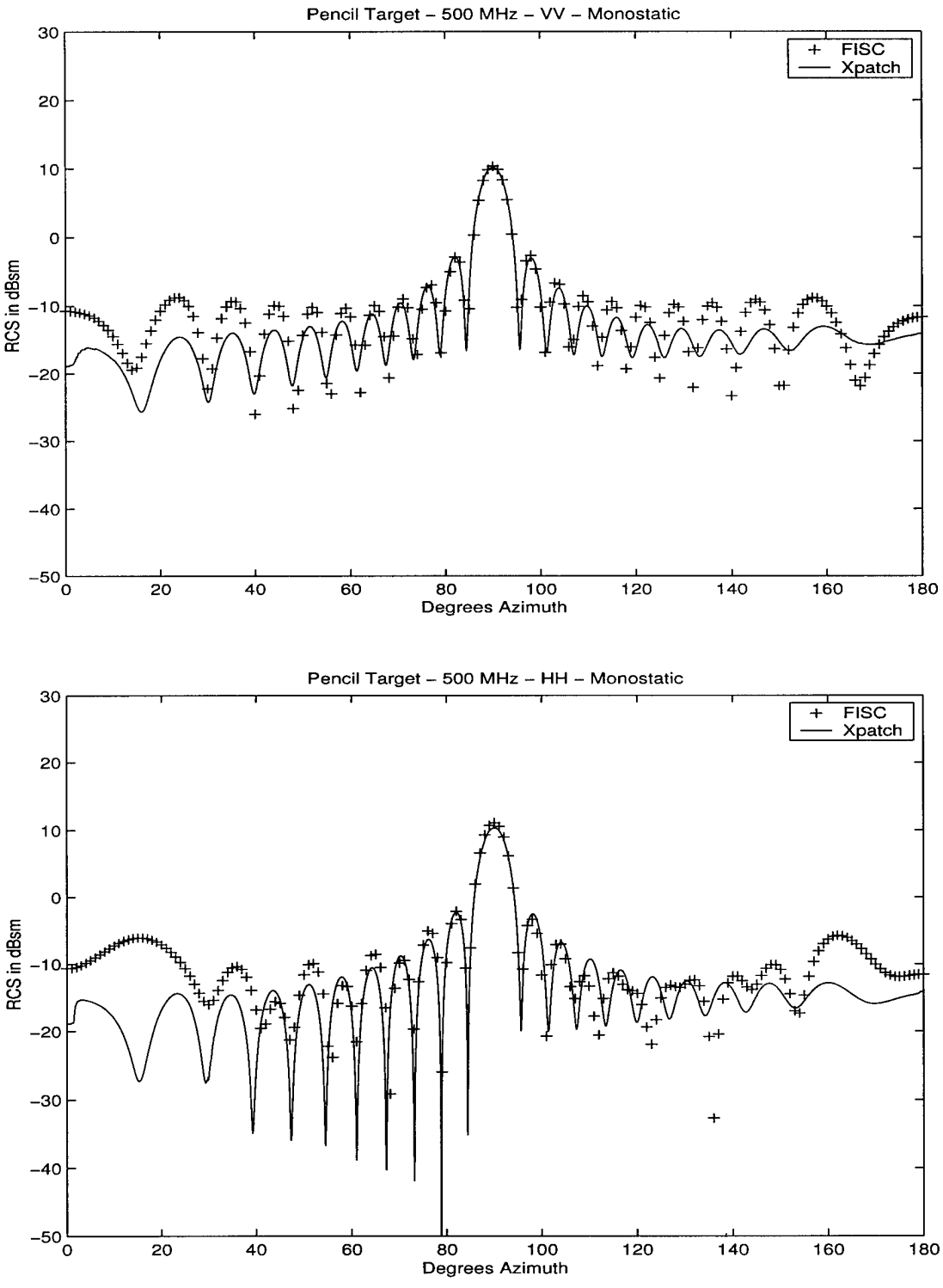

Figure A.1 Monostatic scattering of pencil at 500 MHz.

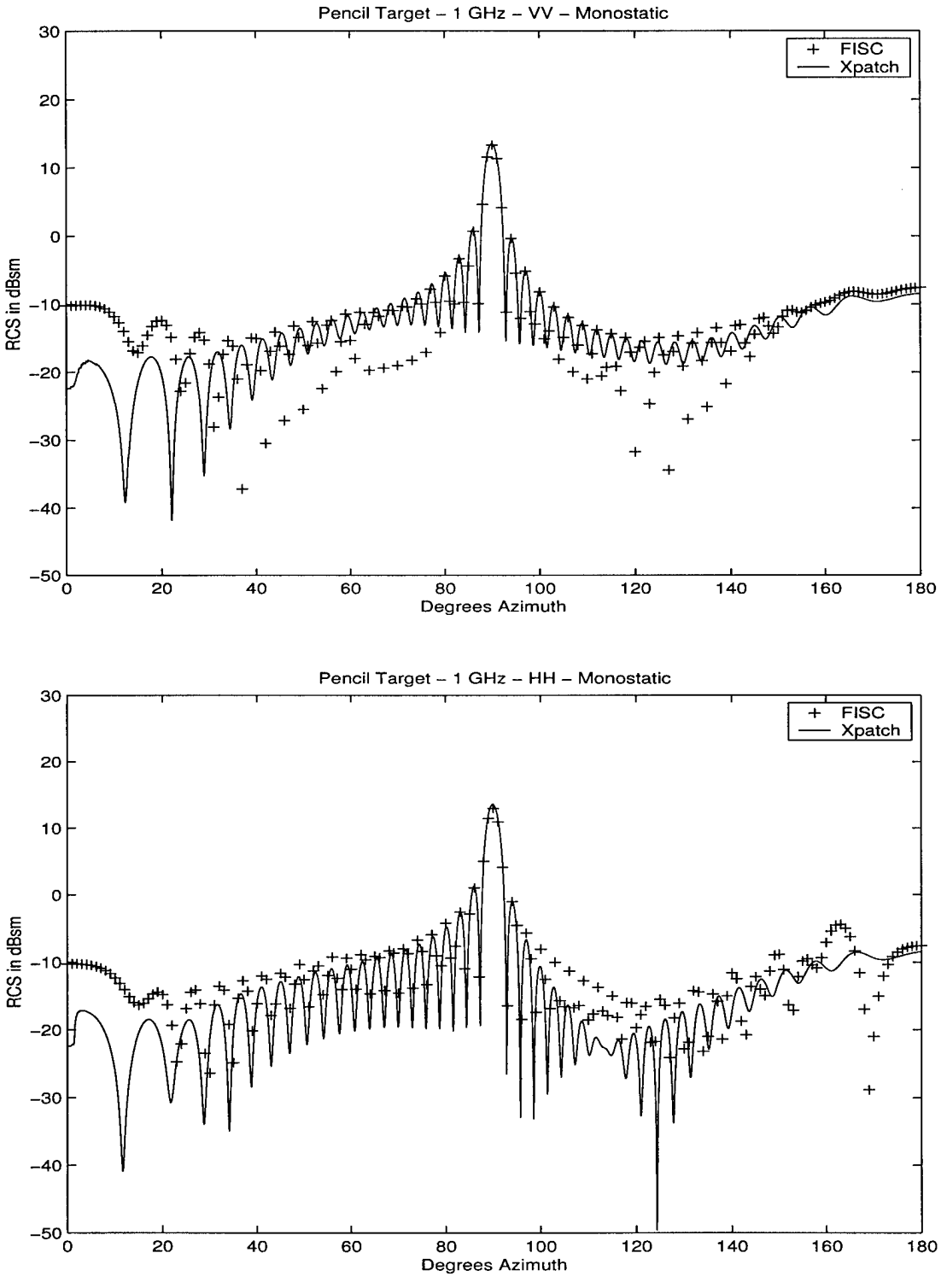

Figure A.2 Monostatic scattering of pencil at 1 GHz.

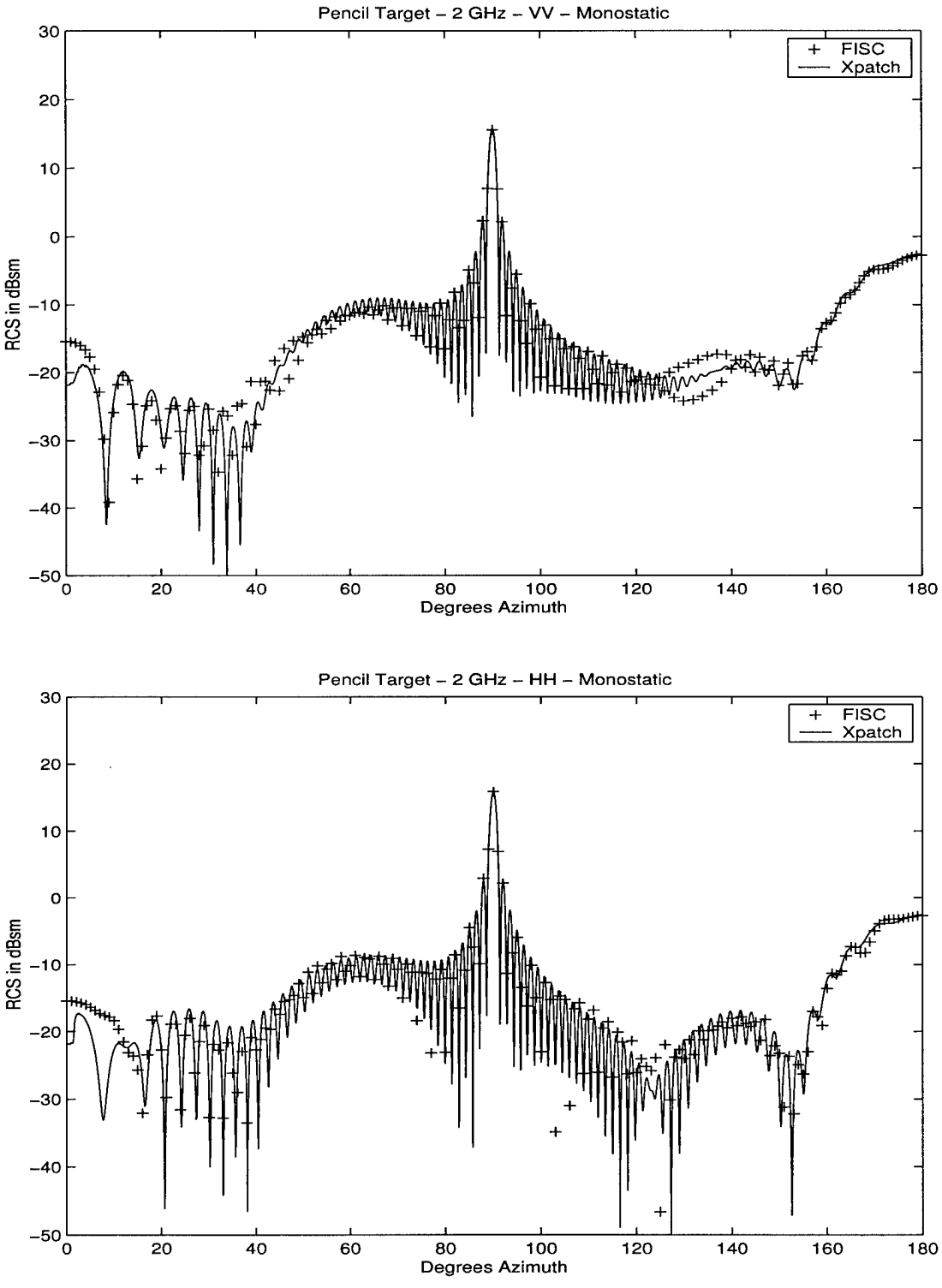

Figure A.3 Monostatic scattering of pencil at 2 GHz.

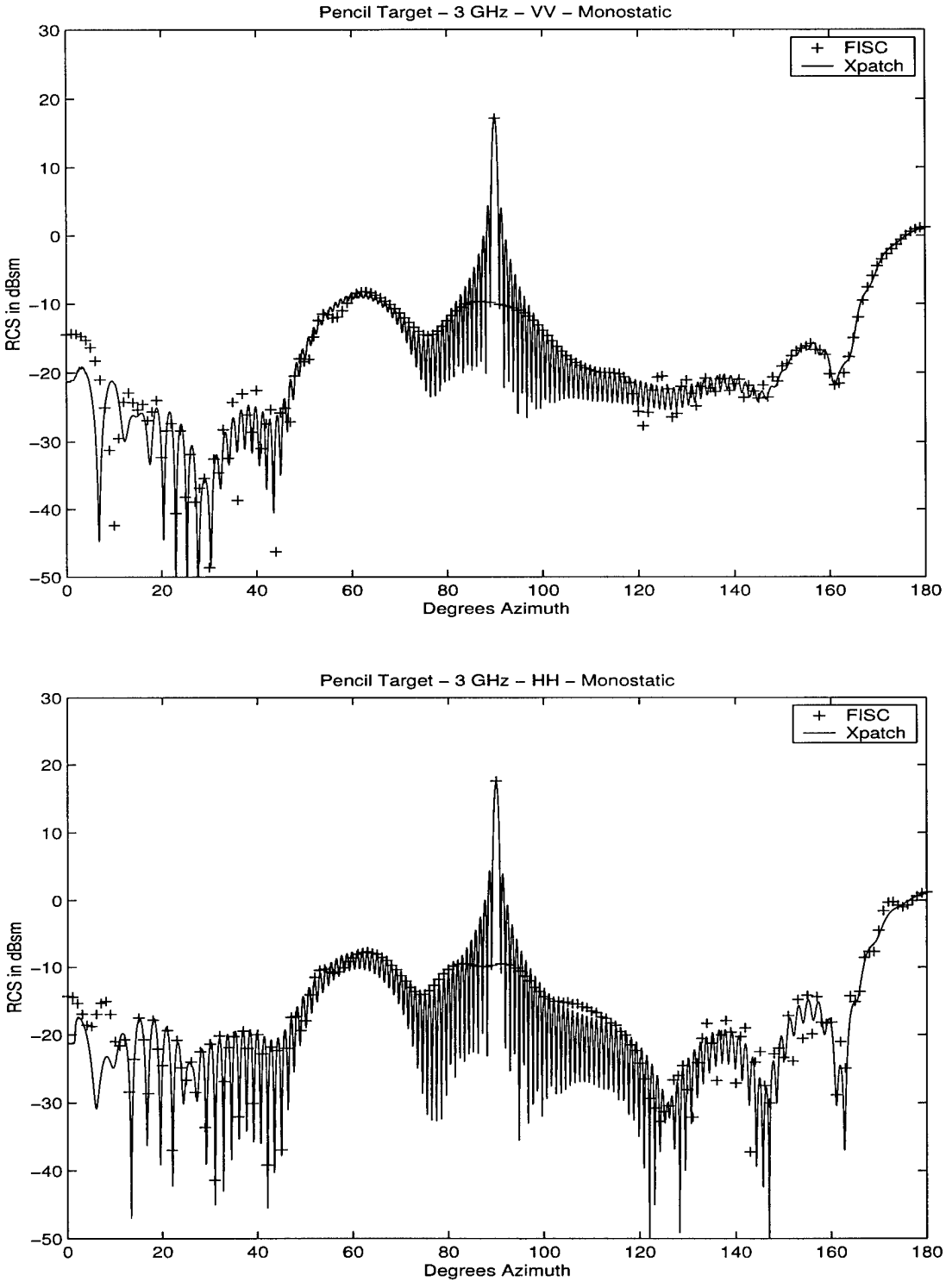

Figure A.4 Monostatic scattering of pencil at 3 GHz.

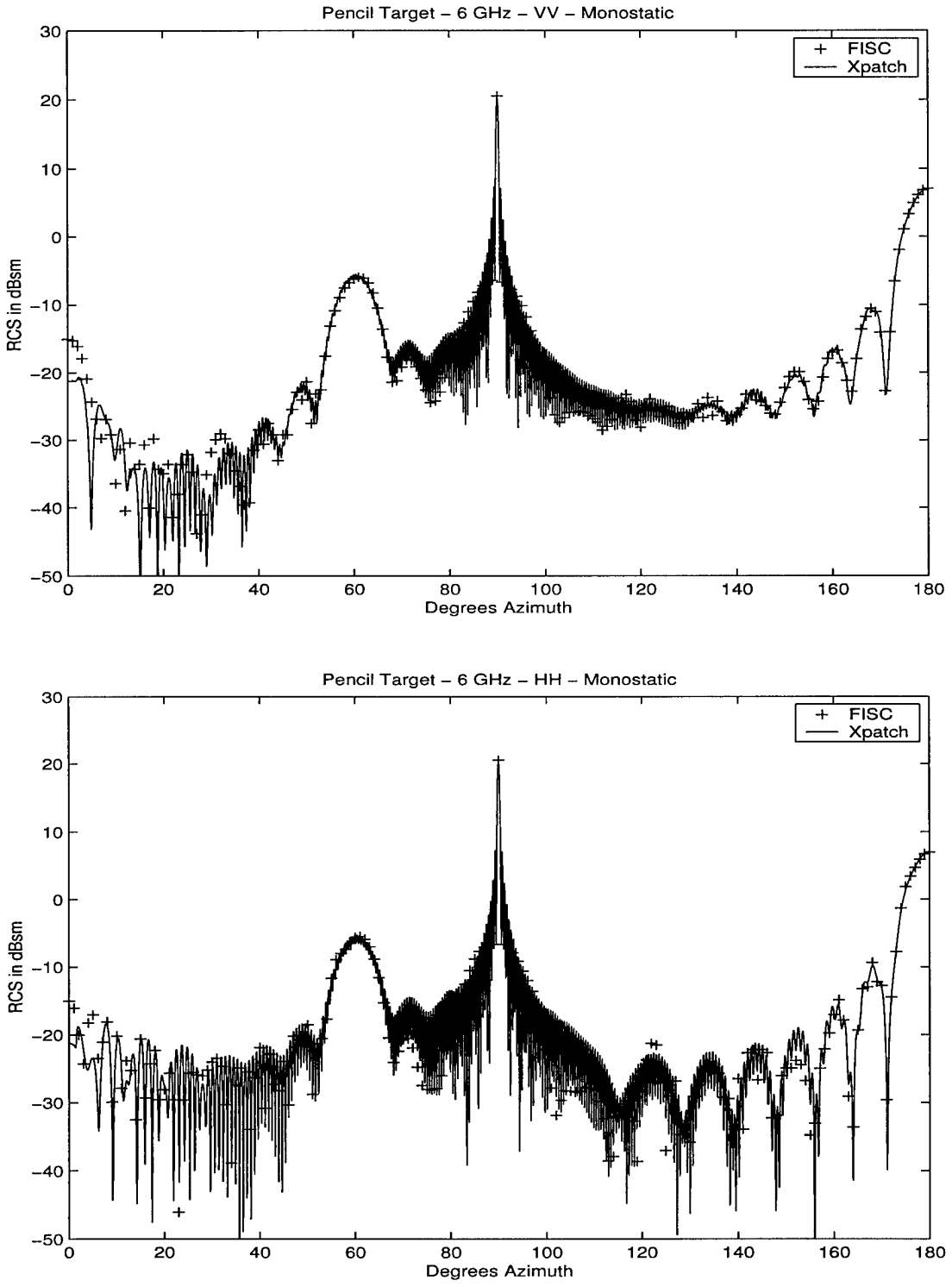

Figure A.5 Monostatic scattering of pencil at 6 GHz.

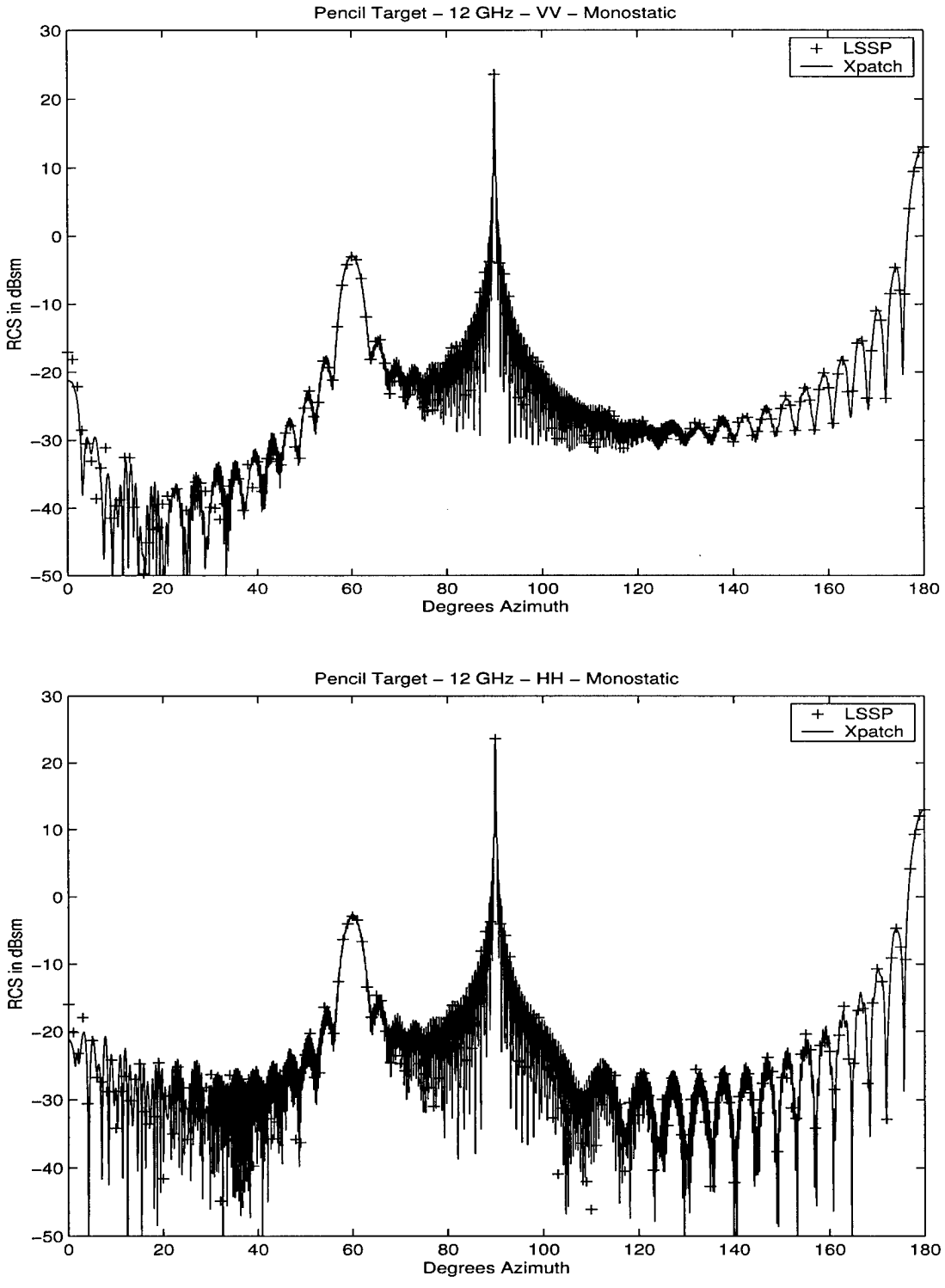

Figure A.6 Monostatic scattering of pencil at 12 GHz.

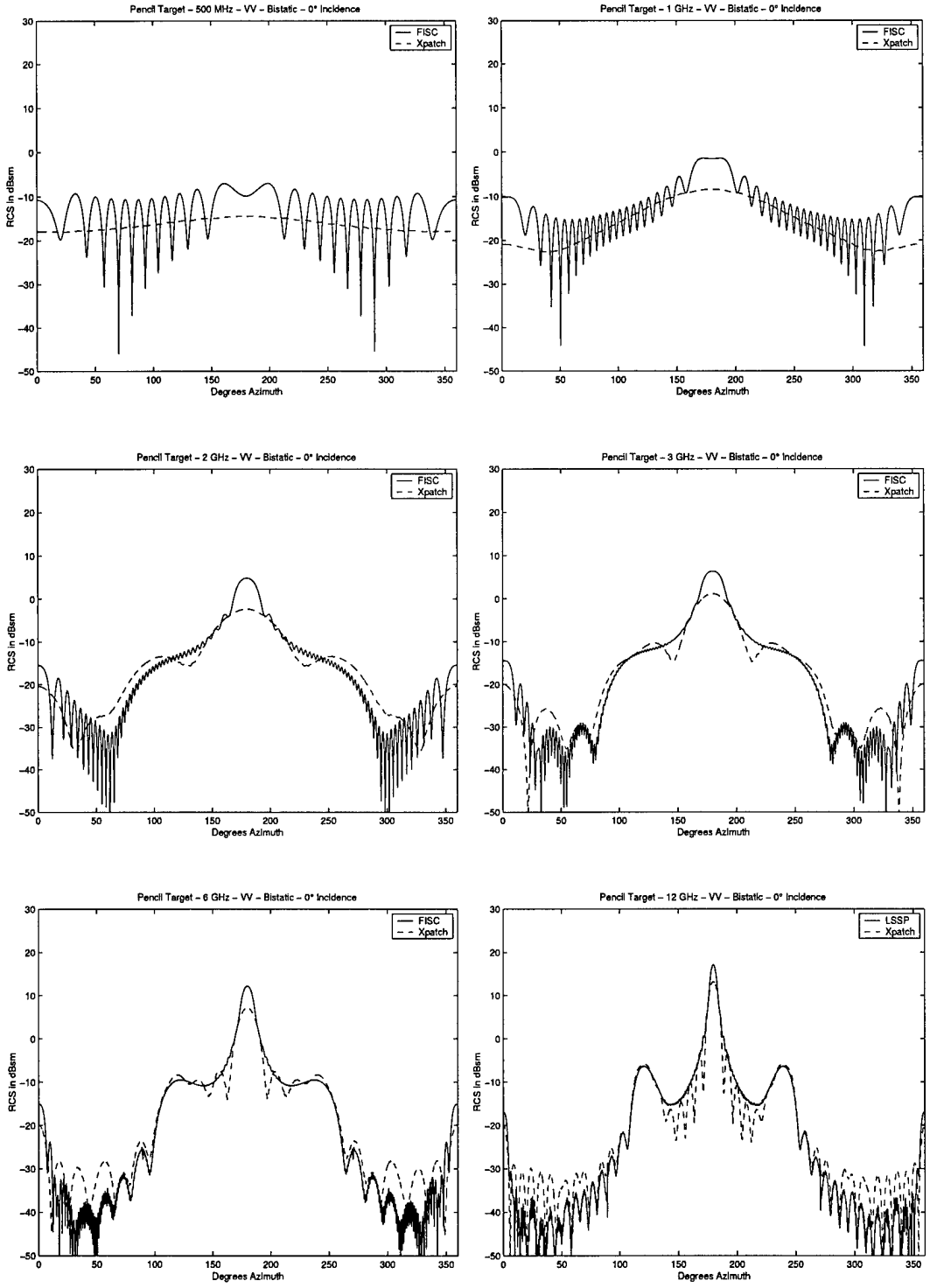

Figure A.7 Bistatic scattering of pencil, VV polarization at 0° incidence. Frequencies are between 500 MHz and 12 GHz.

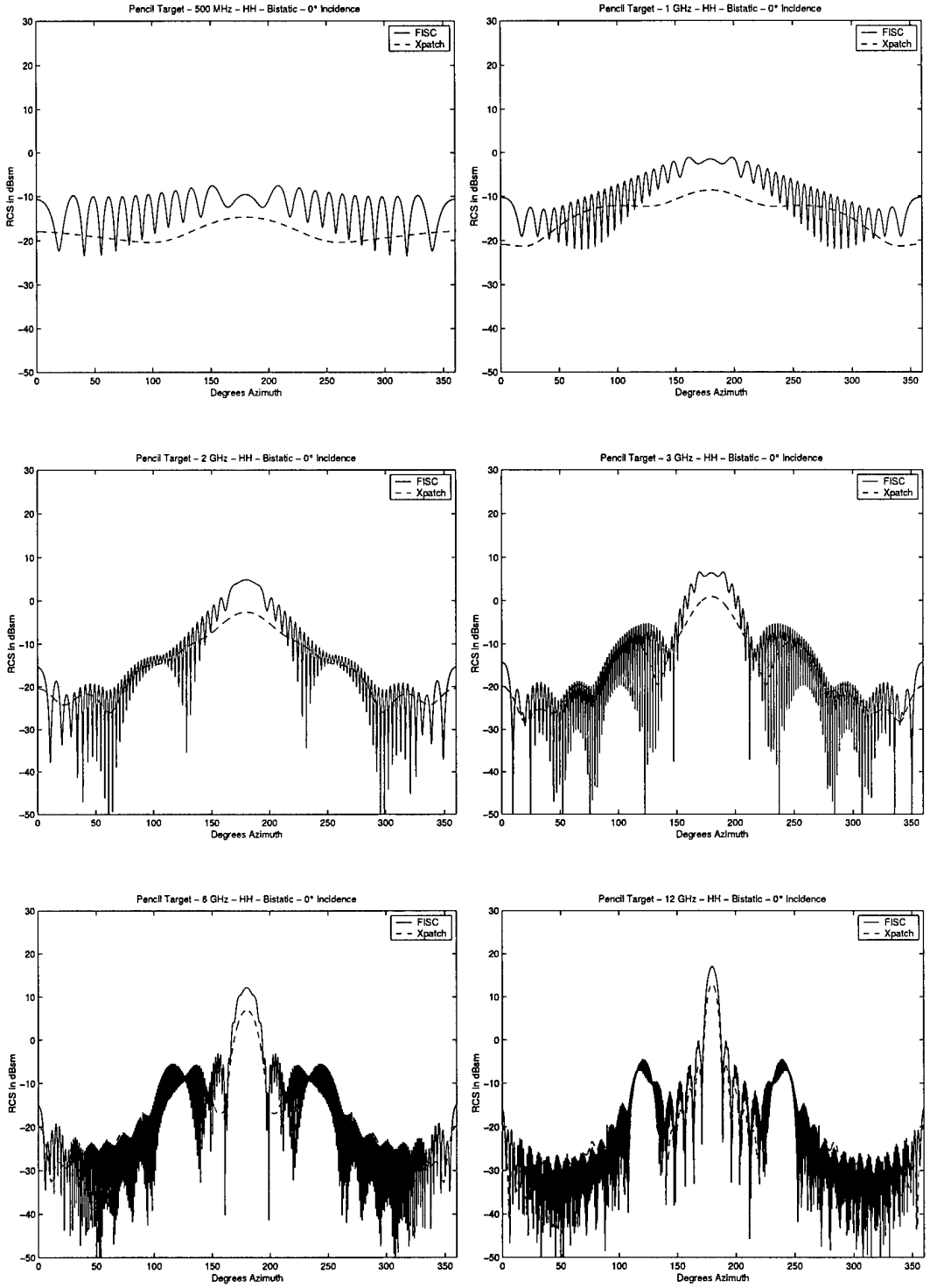

Figure A.8 Bistatic scattering of pencil, HH polarization at 0° incidence. Frequencies are between 500 MHz and 12 GHz.

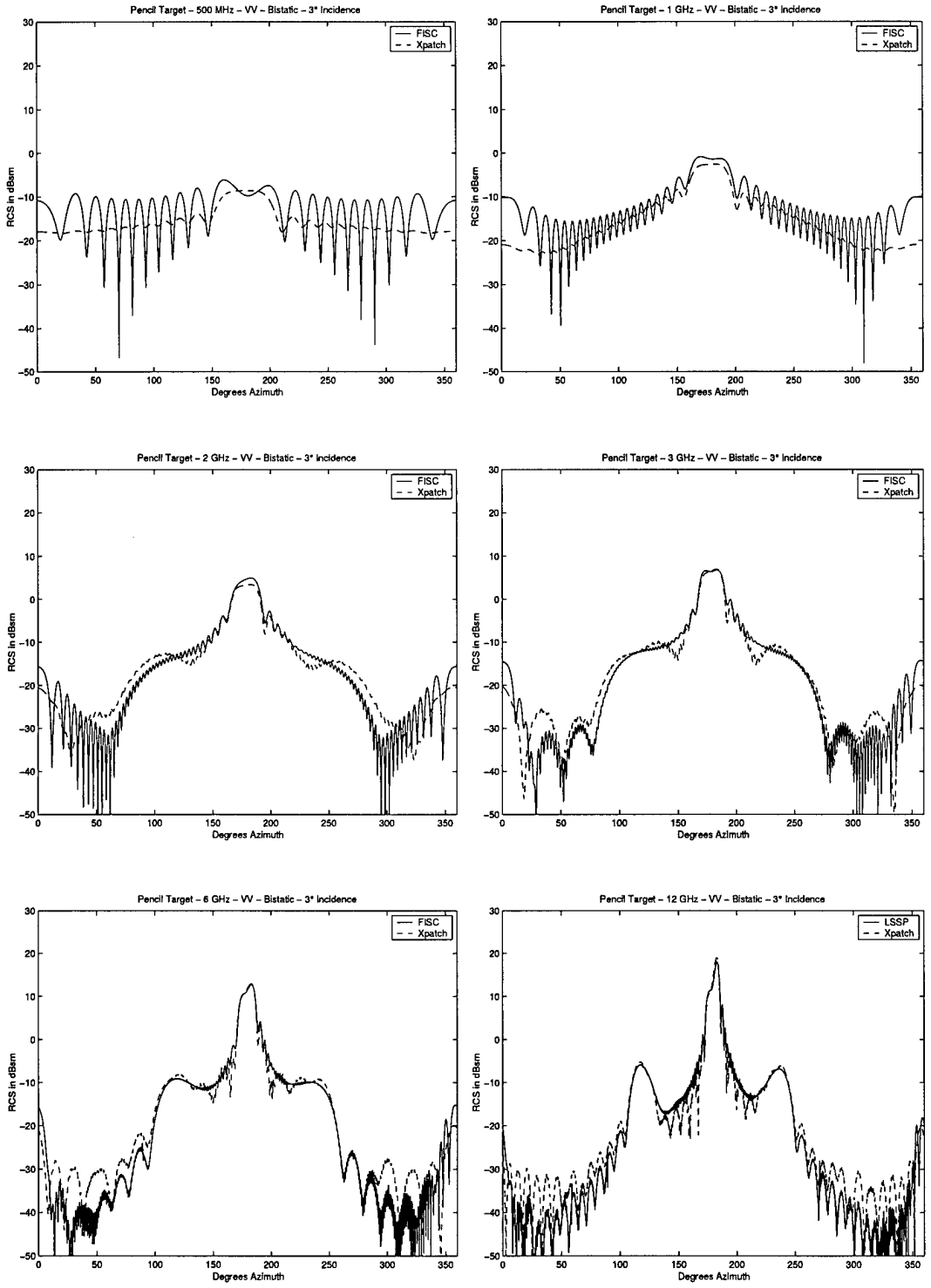

Figure A.9 Bistatic scattering of pencil, VV polarization at 3° incidence. Frequencies are between 500 MHz and 12 GHz.

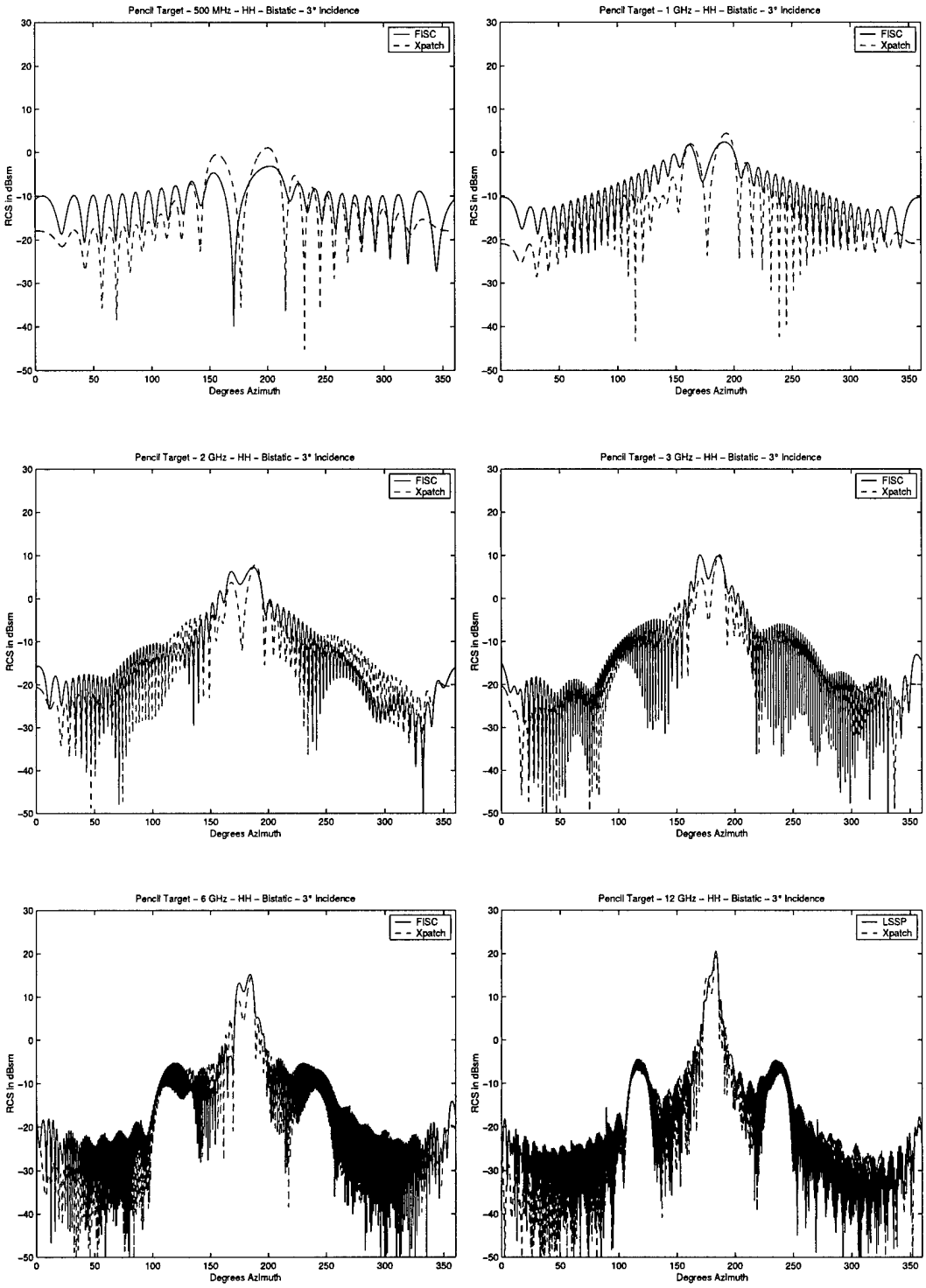

Figure A.10 Bistatic scattering of pencil, HH polarization at 3° incidence. Frequencies are between 500 MHz and 12 GHz.

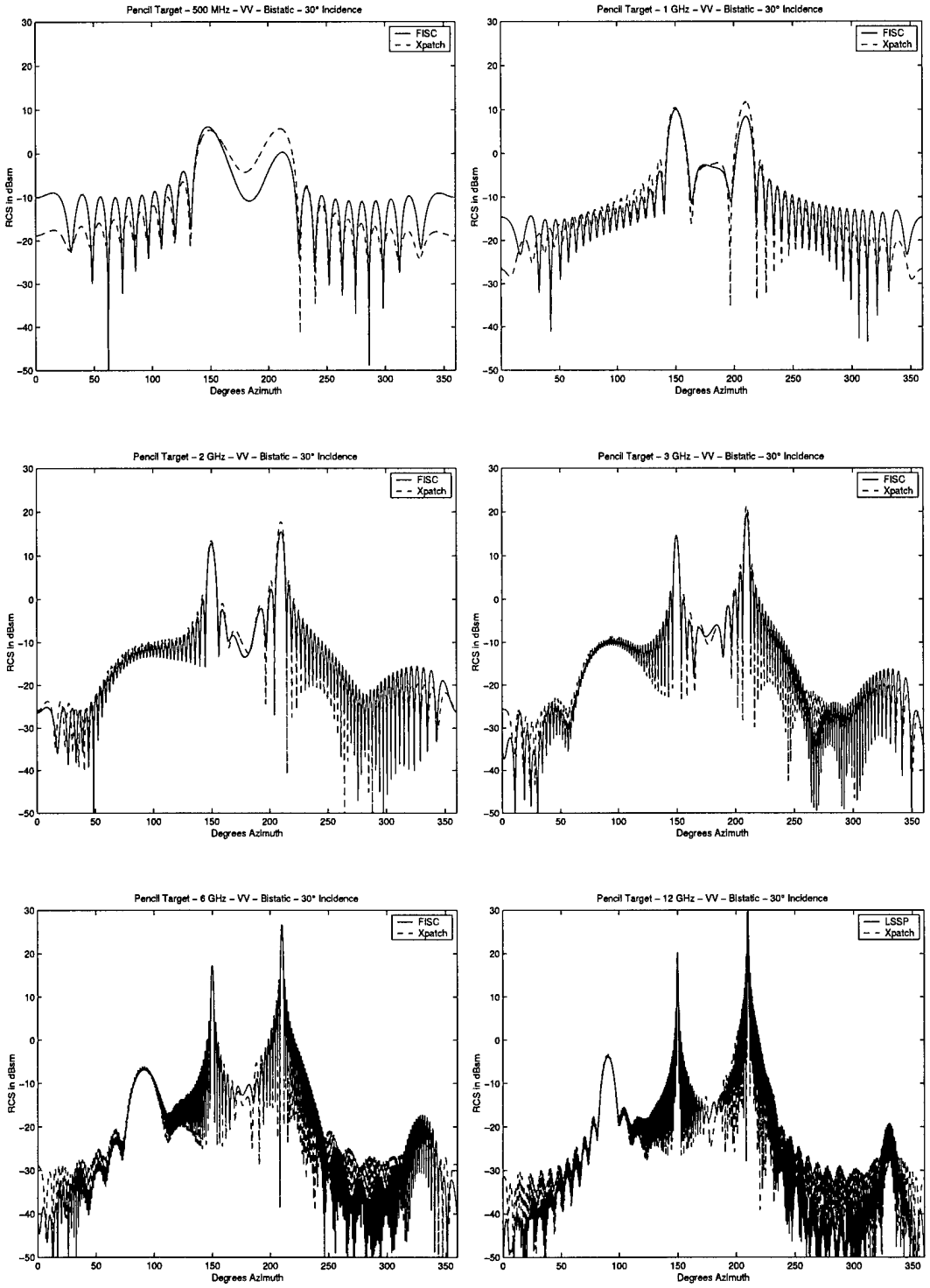

Figure A.11 Bistatic scattering of pencil, VV polarization at 30° incidence. Frequencies are between 500 MHz and 12 GHz.

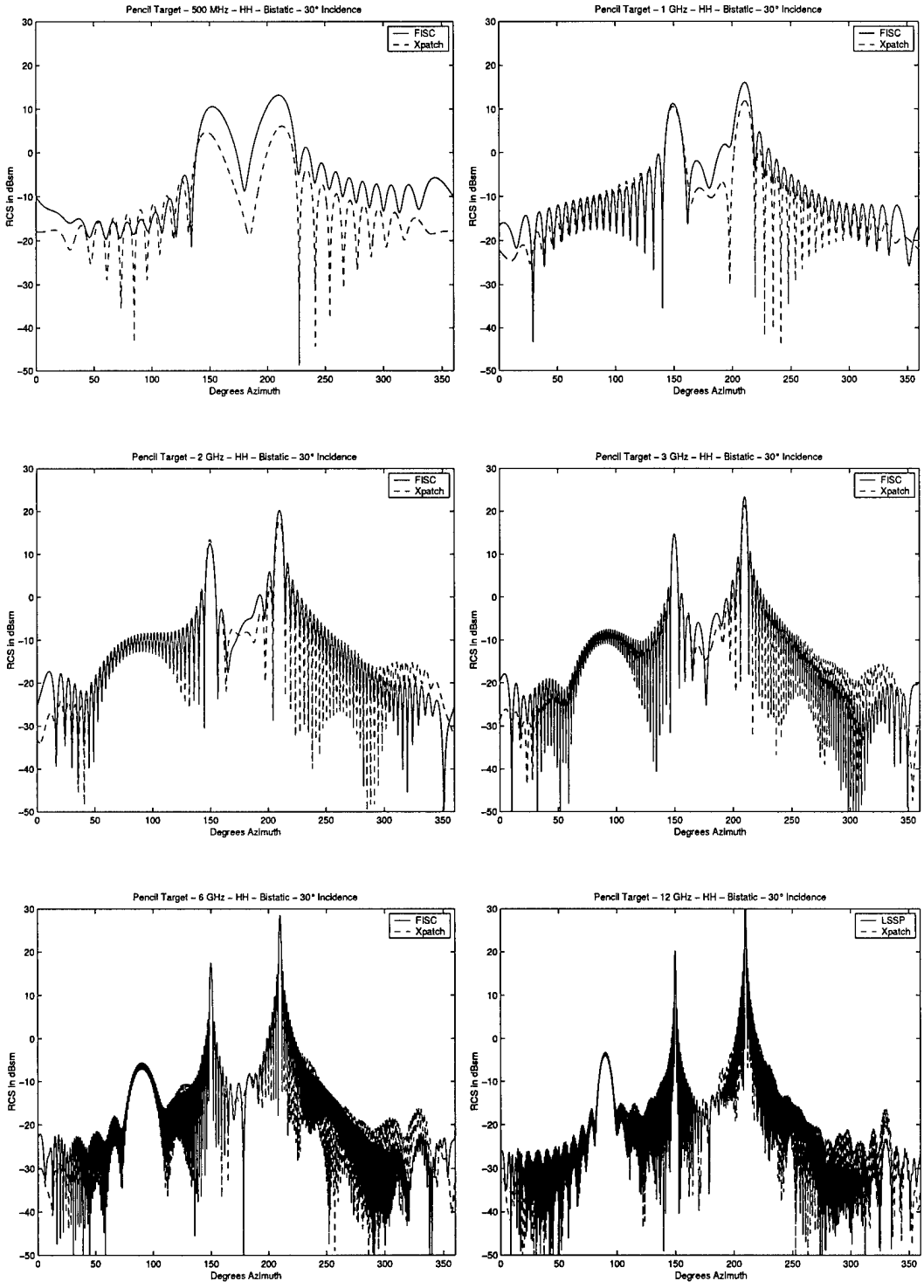

Figure A.12 Bistatic scattering of pencil, HH polarization at 30° incidence. Frequencies are between 500 MHz and 12 GHz.

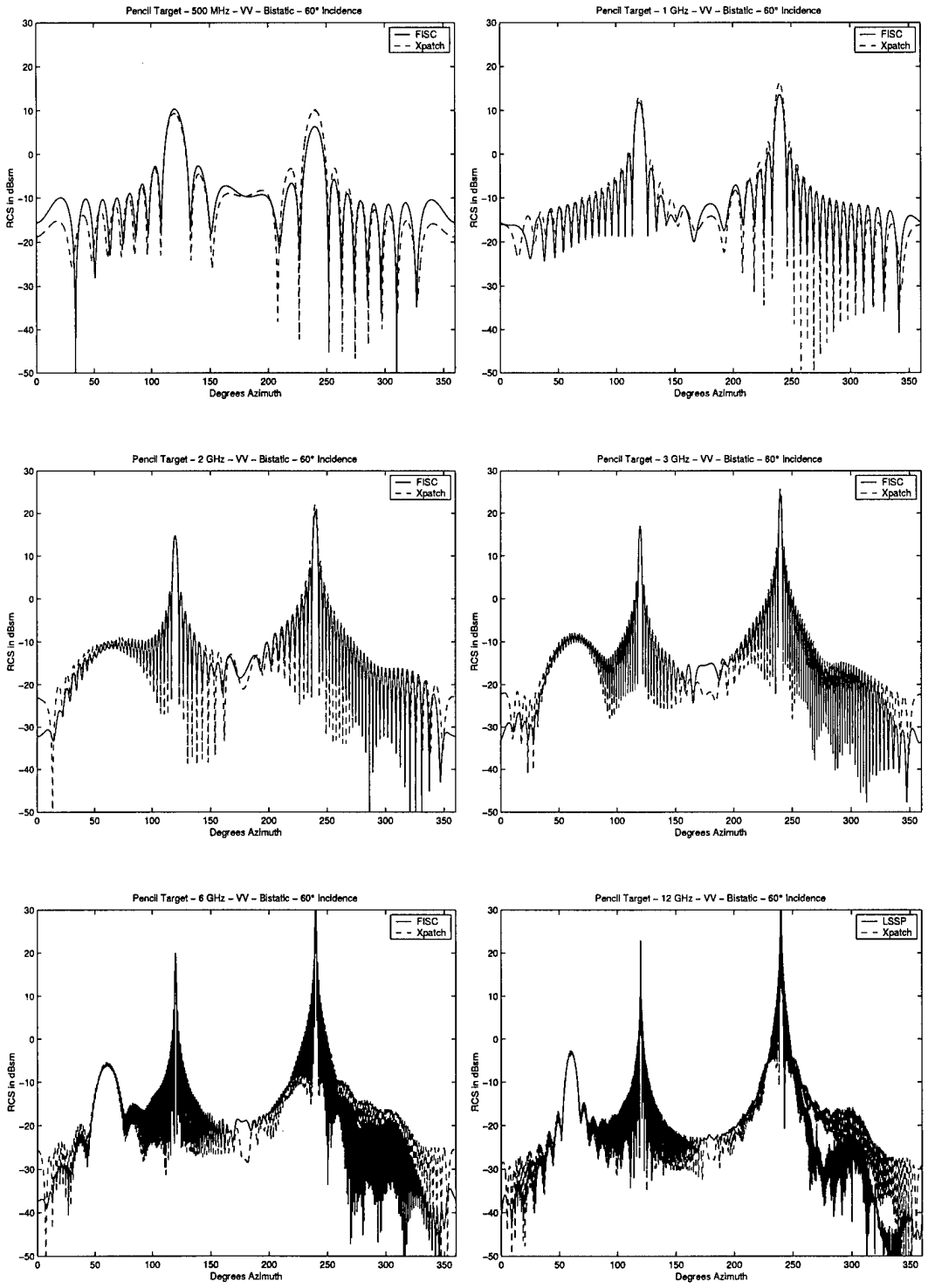

Figure A.13 Bistatic scattering of pencil, VV polarization at 60° incidence. Frequencies are between 500 MHz and 12 GHz.

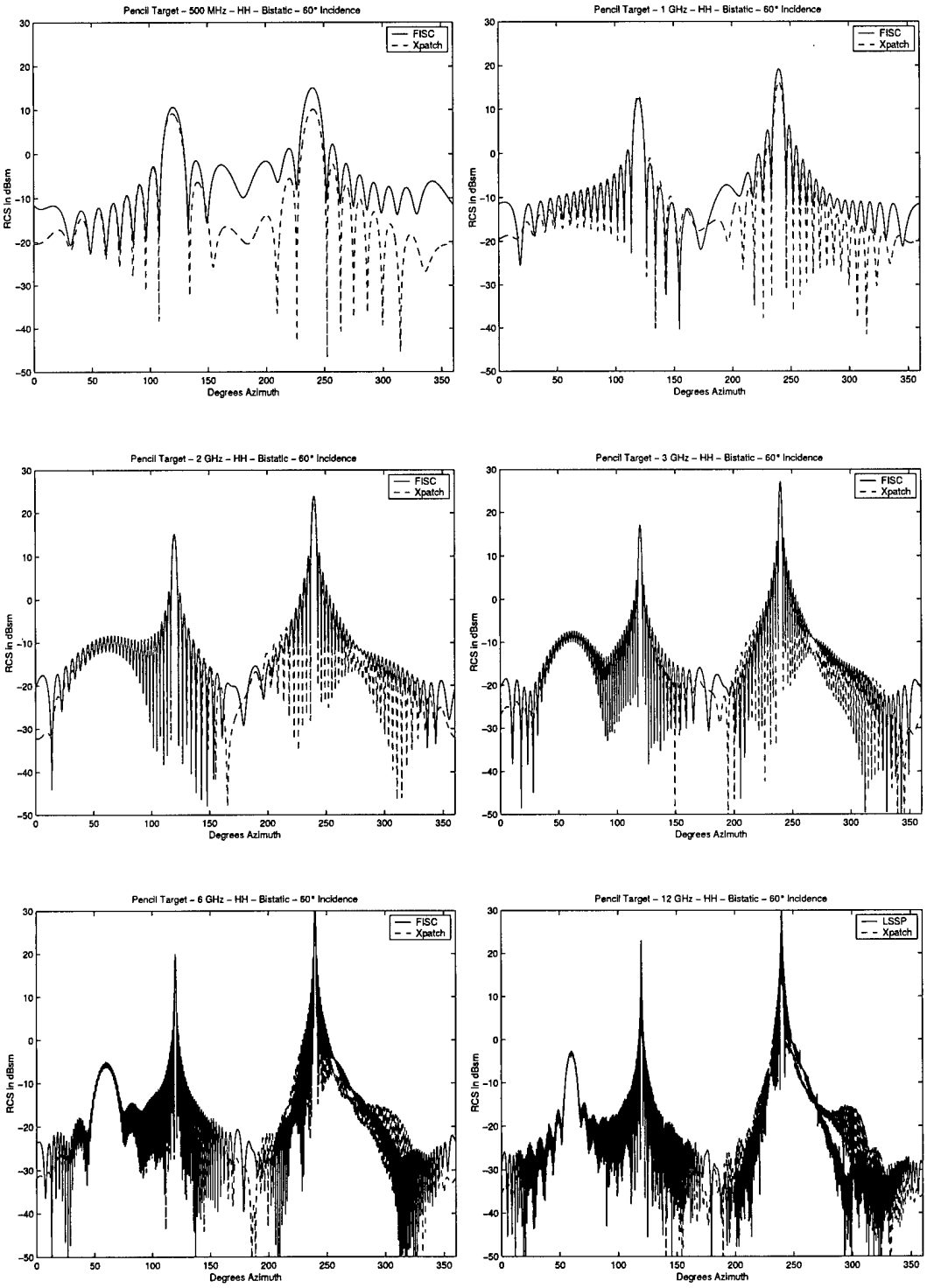

Figure A.14 Bistatic scattering of pencil, HH polarization at 60° incidence. Frequencies are between 500 MHz and 12 GHz.

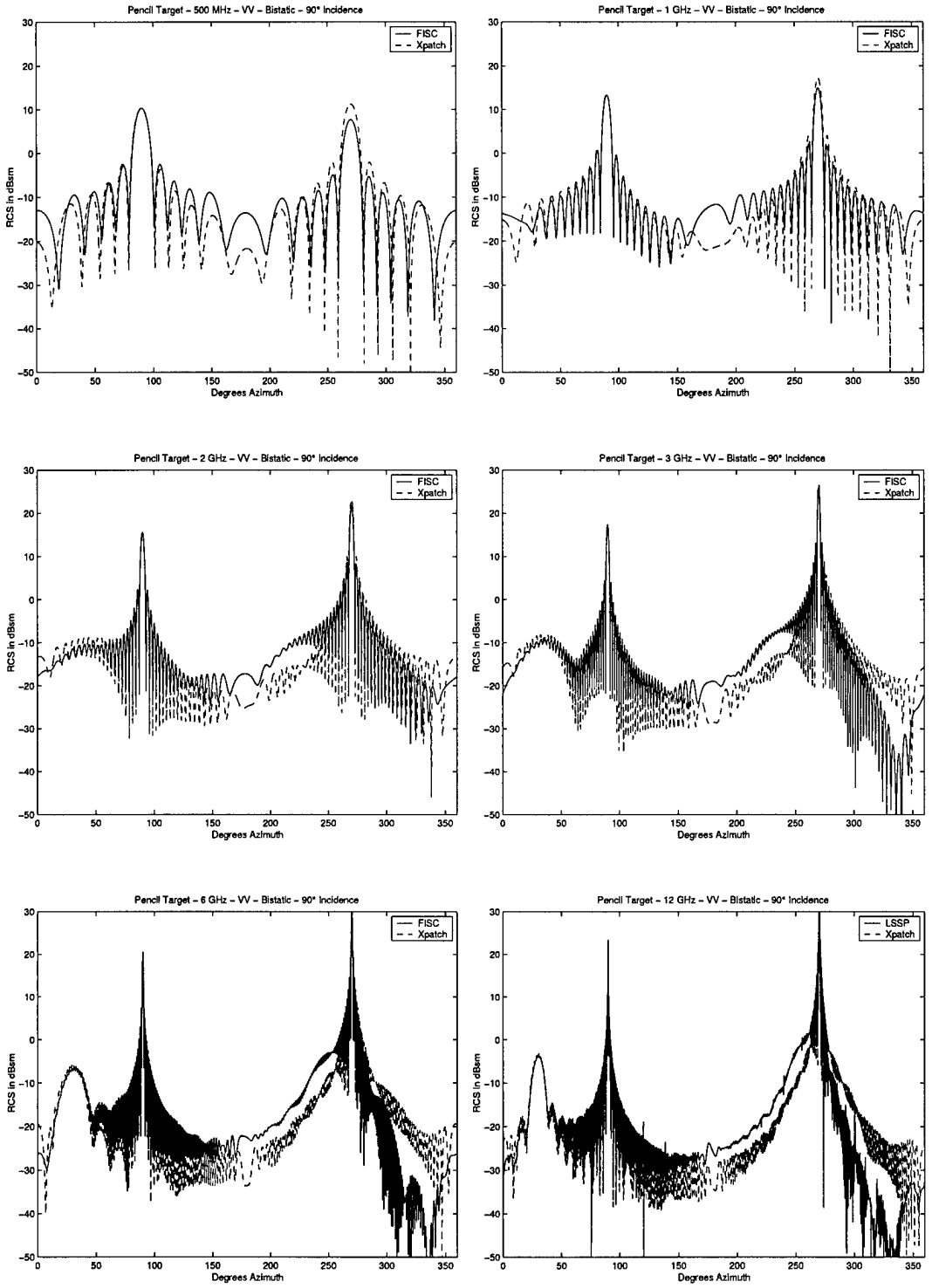

Figure A.15 Bistatic scattering of pencil, VV polarization at 90° incidence. Frequencies are between 500 MHz and 12 GHz.

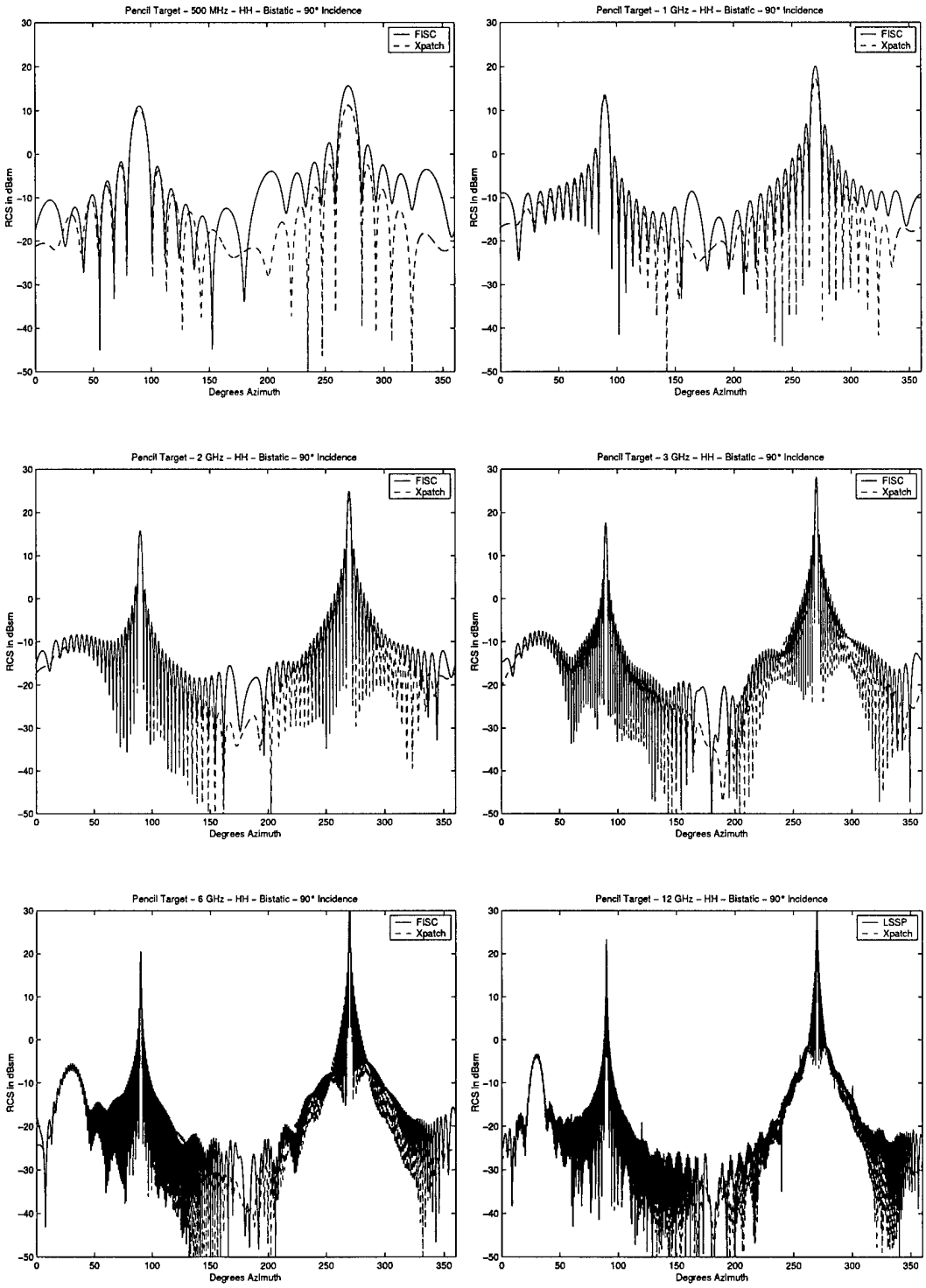

Figure A.16 Bistatic scattering of pencil, HH polarization at 90° incidence. Frequencies are between 500 MHz and 12 GHz.

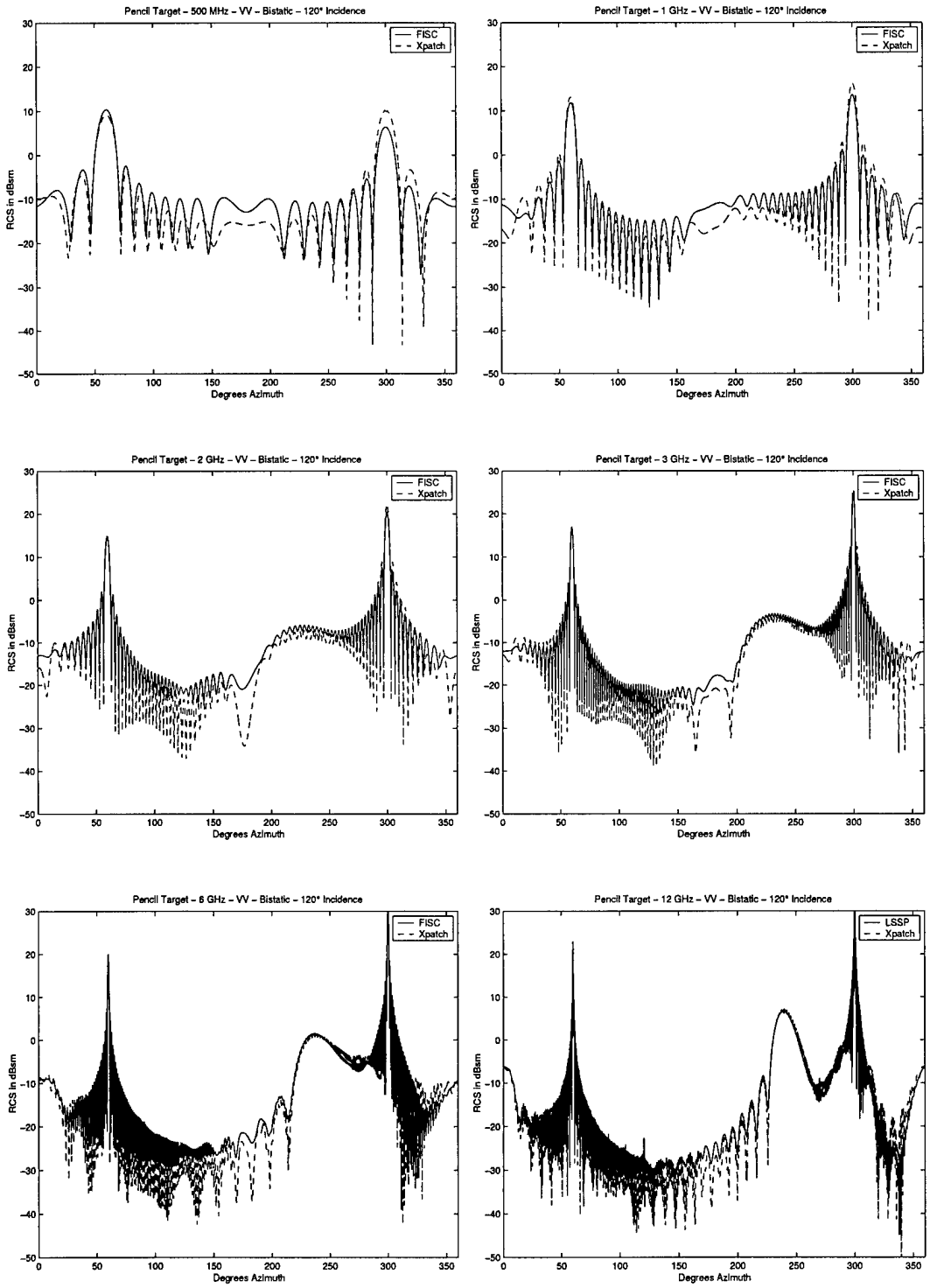

Figure A.17 Bistatic scattering of pencil, VV polarization at 120° incidence. Frequencies are between 500 MHz and 12 GHz.

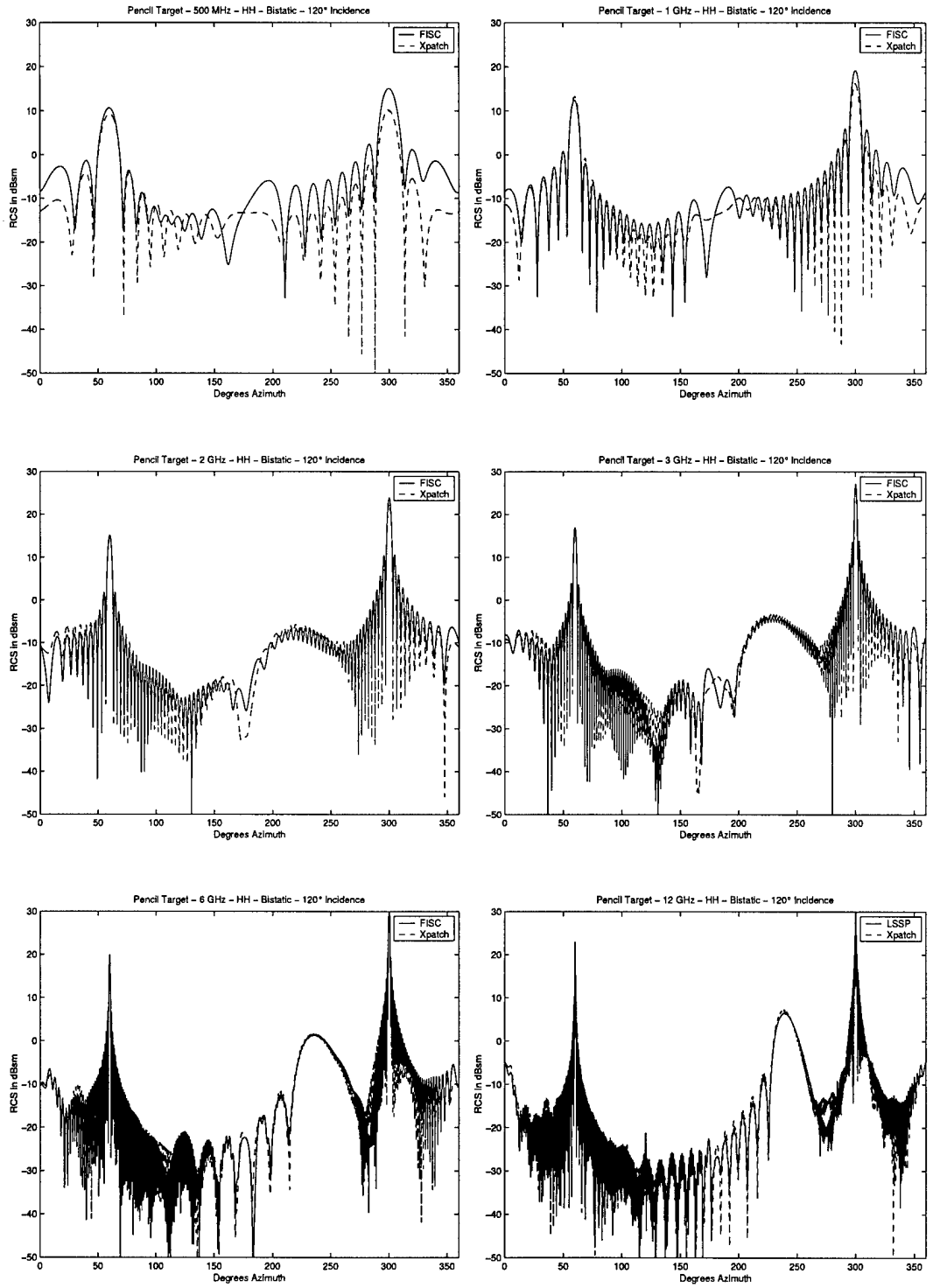

Figure A.18 Bistatic scattering of pencil, HH polarization at 120° incidence. Frequencies are between 500 MHz and 12 GHz.

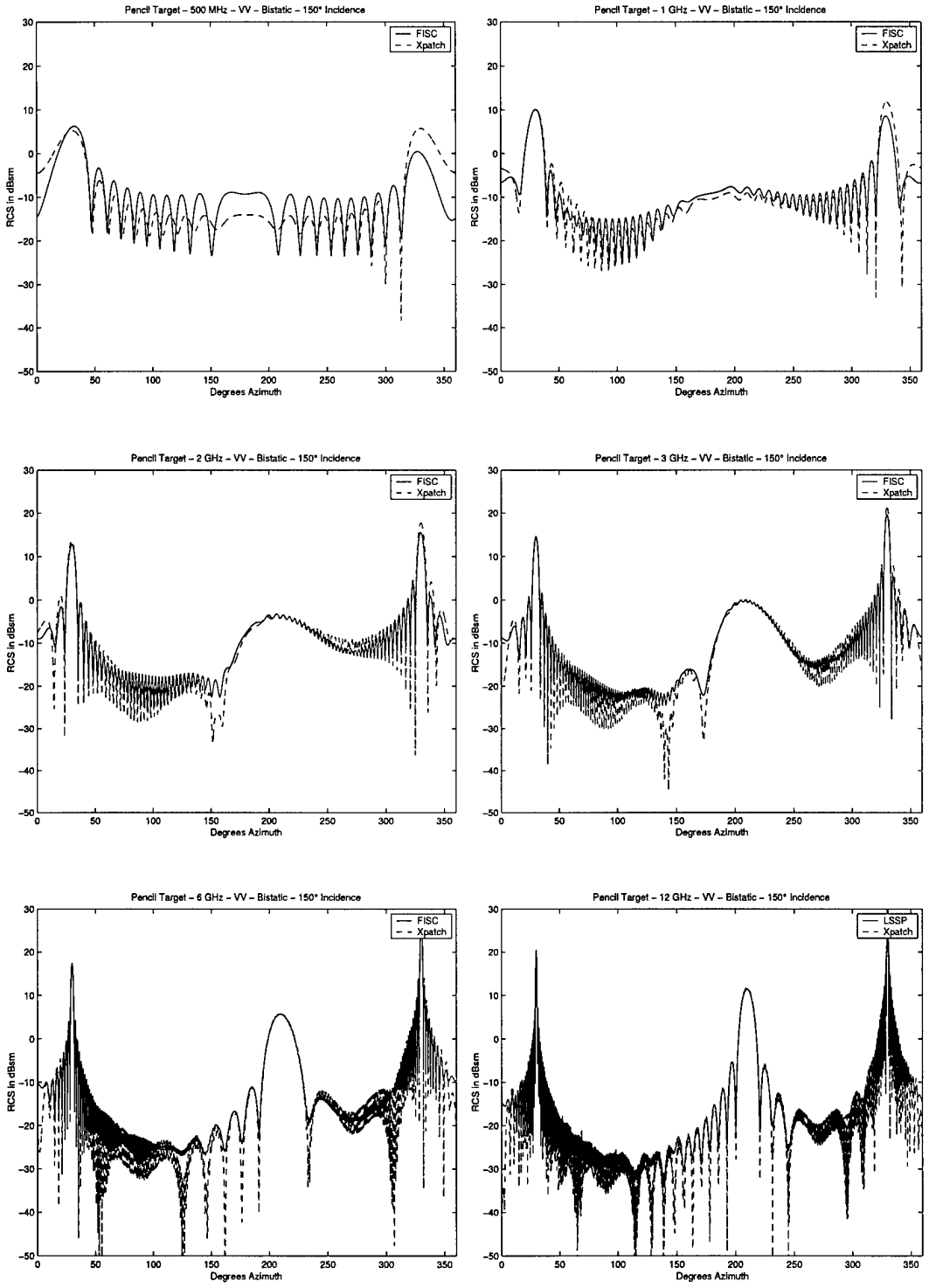

Figure A.19 Bistatic scattering of pencil, VV polarization at 150° incidence. Frequencies are between 500 MHz and 12 GHz.

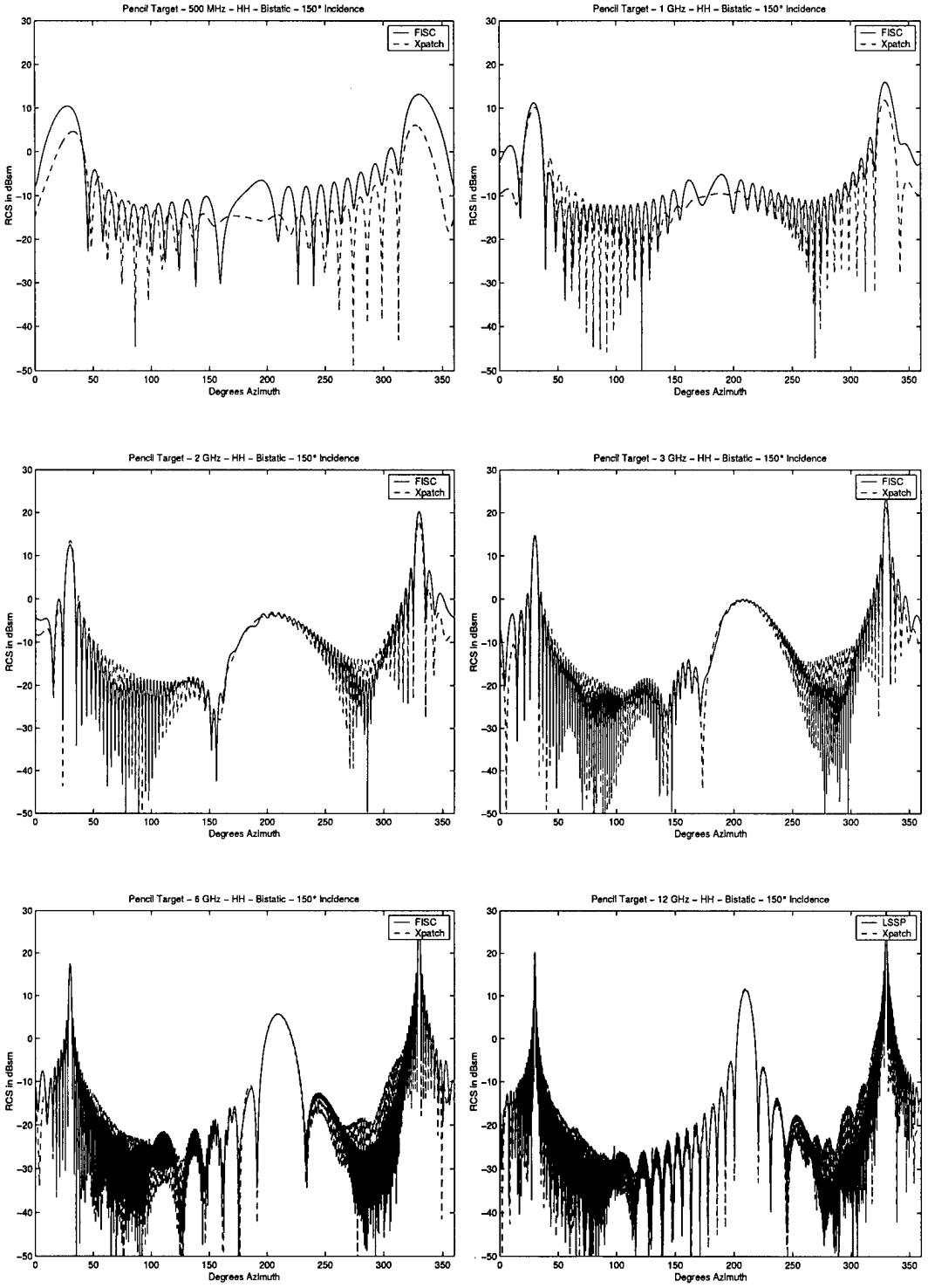

Figure A.20 Bistatic scattering of pencil, HH polarization at 150° incidence. Frequencies are between 500 MHz and 12 GHz.

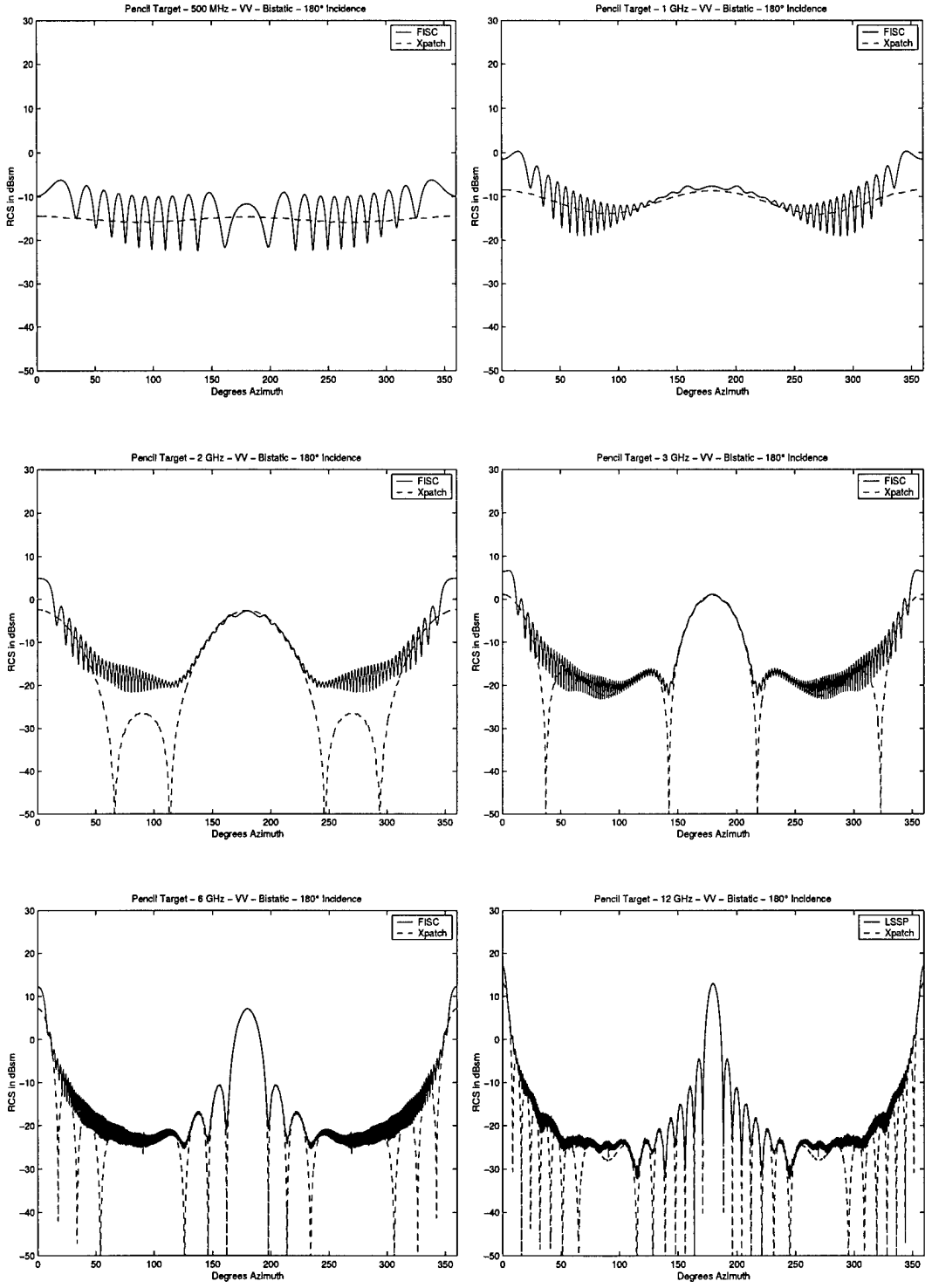

Figure A.21 Bistatic scattering of pencil, VV polarization at 180° incidence. Frequencies are between 500 MHz and 12 GHz.

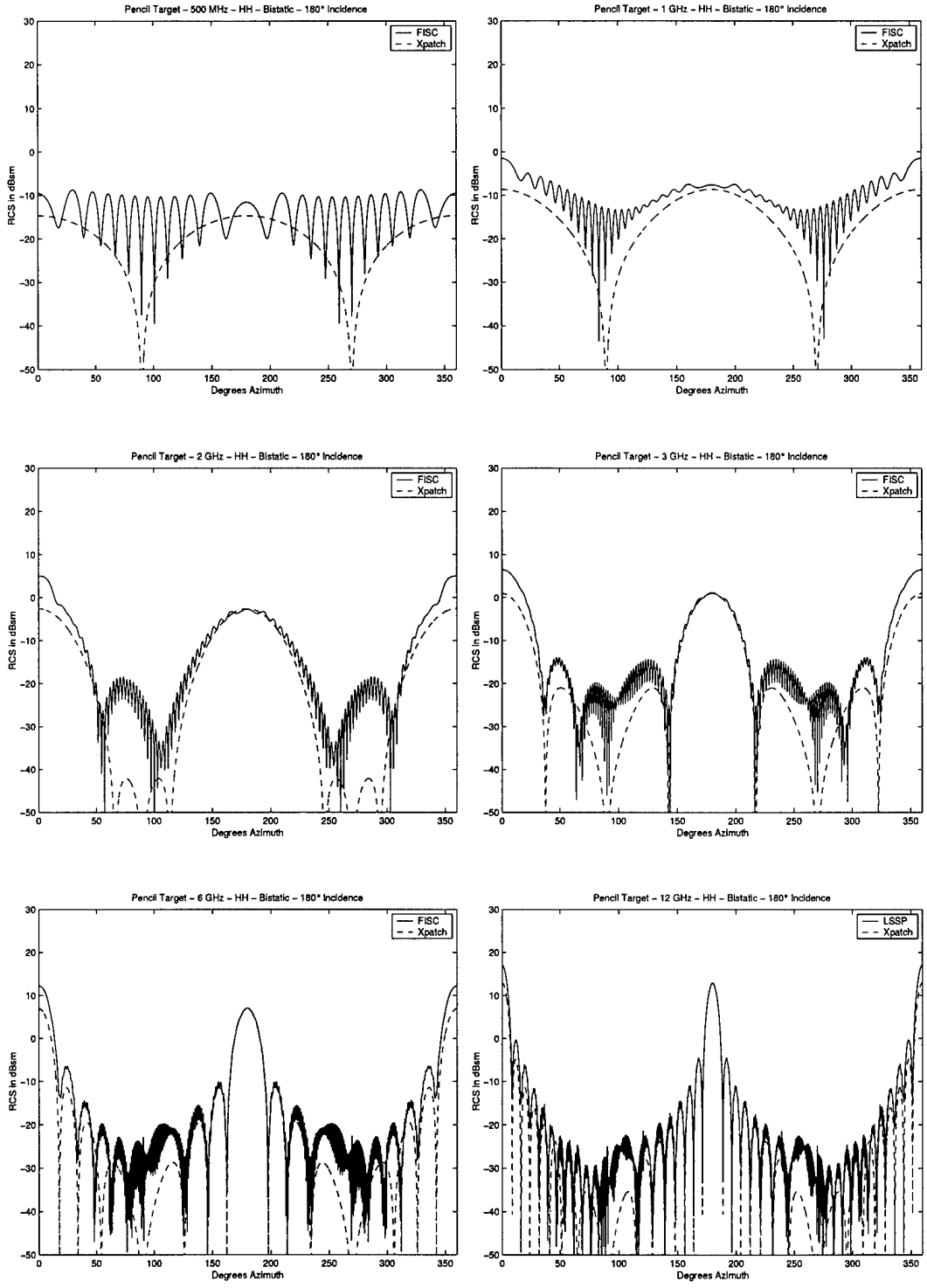

Figure A.22 Bistatic scattering of pencil, HH polarization at 180° incidence. Frequencies are between 500 MHz and 12 GHz.

APPENDIX B

NOTES ON LSSP

B.1 Getting Started with LSSP

The theory and implementation of parallel MLFMA require a tremendous background of understanding and computer coding skills. However, running the large-scale scattering program, LSSP, requires significantly less understanding. This appendix is designed to provide enough details for an engineer to run the code and begin to understand the process required to install the code through data acquisition. Before compiling the code, an understanding of the code structure is necessary and described in the next section. It is followed by a discussion of the message passing interface software required for installation and other setup factors.

B.1.1 LSSP code structure

LSSP has two distinct and structurally different parts. It is the marriage of the ScaleME library, which acts as the FMA kernel plus the TRIMOM code for the MoM calculations. The main program is compiled and linked to the ScaleME library that is created under the fma2 directory. The main program is located under the src directory under the path of progs/lssp/ along with other utility programs. The utility programs include acad2acad.c, flipNormal.c, pat2edg.c, acada2b.c, and mkcenters.c. They are used to process the geometry file. Many problems can occur if the geometry file is incorrect. Using the ACAD format to describe the geometry, acad2acad converts all parts into a single large part and has the ability to change the units from meters to inches. The routine flipNormal is used to reverse the normals as a closed object with normals facing inward cannot be solved by LSSP. This was found when first testing the pencil target that was originally made with inward-facing facet normals. From the output file of acad2acad, pat2edg creates the edge file. Using the edge file and the processed facet file, acada2b simply combines the information into a binary format for rapid reading and smaller size. Once these steps are performed once on a target, they need not be performed again.

With processed geometry information in binary format, mkcenters divides out the edges based on location and load balancing and how many processors will be used. These center files must be recreated if a different number of processors is used for a run.

The other routines are associated with main.c source code and include scaleme.c, readData.c, postlssp.c, cenArray.c, conjgSymm.c, and ftoc.c on the C-side. FORTRAN subroutines on the lssp side are cgetrf.f, cgetri.f, cgmres.f, and trimom.f. The scaleme.c file is used for initialization and finalization, and it outputs some timing information. The geometry and input file data are read in using readData.c code. After a run is completed and has been iterated to an acceptable residual error, postlssp.c has the functions to produce readable output data. There are also several scripts used to run jobs as well as post-process the log files. The key script for running the jobs is lssp.pl and to look at the tree structure and memory usage from the log file, the script lsspinfo.pl is used. The latter script has problems with reporting memory on large-scale problems because of integer overflow errors. The best thing to mitigate this problem is to use a spreadsheet to calculate the precise amount of memory based on the information in Chapter 7.

B.1.2 Message passing interface

ScaleME requires the message passing interface (MPI). MPI is usually installed on supercomputers or clusters of machines that are networked for parallel processing. If MPI is not installed on a system, it can be easily downloaded from the website *http://www-unix.mcs.anl.gov/mpi/*, which also describes the MPI standards. MPICH is a portable implementation of the MPI standards. Once the software is downloaded, it must be configured through a provided script and then must be compiled. For local runs on the SUN cluster (see Figure B.1) the configuration script must include the flag –rsh=ssh.

B.1.3 Secure shell

The secure shell (SSH) is used to connect to individual nodes via the command, mpirun. SSH must be configured with the appropriate key generations and a passphrase to avoid typing ones password over insecure lines. If the passphrase is left blank, starting jobs is much easier, and each process will not require typing in a passphrase. It does not require too many runs to get the idea that waiting to type a passphrase for each process being run on each node is inefficient.

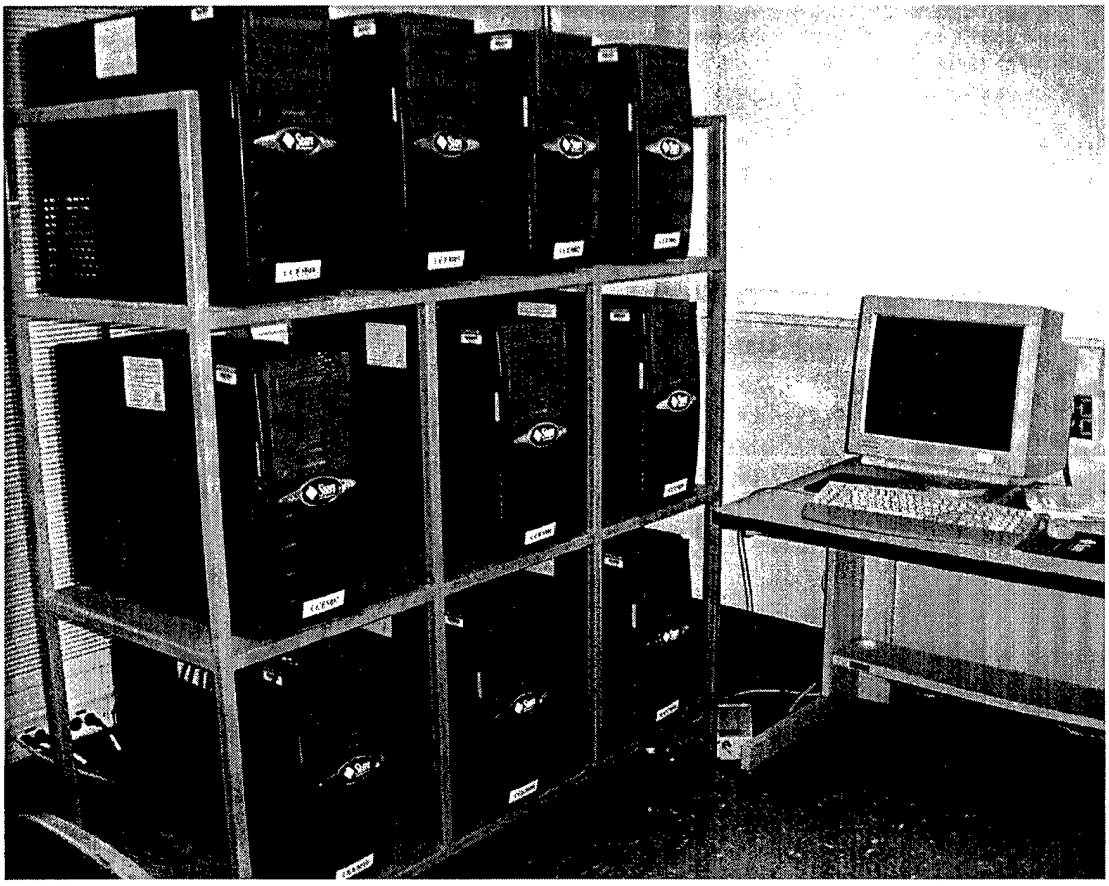

Figure B.1 CCEM SUN Blade 2000 cluster.

B.1.4 Environment variables

ScaleME requires careful usage of environment variables in order to compile and run. These are set up by modifying a file named .scaleme and sourcing it. They will set the paths to the directory structure when compiling and running.

B.1.5 Concurrent versions system

CVS is a version control system that allows storage and tracking of the source code. Any developer should become acquainted with its capabilities and usage. It will rescue the developer that unintentionally or intentionally makes changes that cause problems compiling or running. Along with CVS, it is good to learn a debugger like SUN Workshop to step through a program line by line. These are tools of the trade for anyone involved in computational electromagnetics.

B.2 Compiling

Compiling LSSP is a fairly easy task when the proper libraries and tools for compiling C and FORTRAN are in one's path. Sourcing the .scaleme file is necessary before making LSSP. The compiler flags should first be edited according to the machine operating system on which the code will be compiled. A # sign is used to comment out the compiler flags that are not required. There are a few steps to compiling LSSP. First, the command 'make clean' is executed in both the fma2 and progs/lssp directories. This will remove old object files that may exist. By typing 'make' under the fma2 directory, the ScaleME library will be created for implementing FMA. Then, under the progs/lssp directory, 'make' is executed again. This uses the 'Makefile' that produces a binary that links the ScaleME library to the MoM code TRIMOM. Before this linking TRIMOM and the other codes in this directory are compiled. The other codes include utilities for processing the geometry and outputting data.

B.3 Running LSSP

Once the code is compiled, the binaries should be placed in the bin directory next to the src directory containing the fma2 and progs/lssp directories. This directory should be in ones path. To test the code, a small sphere problem called stdTest should be run. The script, lssp.pl, takes care of all the geometry processing and setup. This script should be reviewed to understand the process to run a job. To run the standard test object, the following command is used: lssp.pl --project=stdTest --np=1. This will process the geometry file into a binary file containing facet and edge information. This file with a .geo extension is used to run on a single processor. If the argument to np is changed, the job will be subdivided onto multiple processors according to the value of np. Always before a job is run, a set of np center files with the extension .ctr are created. For large-scale problems, setting up the geometry may be desireable in advance and can be done by adding the flag --geometryonly. If the geometry file is already processed, the --runonly flag is tacked on to the end of the aformentioned command. The data is output to a .fld file which is converted to a .rcs file using postlssp. The .rcs file is an ASCII file representing the binary .fld file.

B.3.1 Input parameters

There are several input parameters that one must become familiar with in order to run LSSP. The frequency, polarization, and aspect angles are easy to understand. LSSP has a high degree of flexibility for the user, so big mistakes are easy if the user does not understand the technology and requirements associated with making runs. For example, the geometry file must be preprocessed to be appropriate for the frequency of operation. The alpha parameter is used to weight the EFIE and MFIE solutions to avoid resonance problems. For closed objects this is usually set to 0.5.

The accuracy of the integration over each triangle is based on the inputs for integration rules. An input of 2 means that at least a four-point Gaussian-quadrature rule will be used. By looking in the source code of TRIMOM, these parameters can be seen. The code is fixed to use GMRES with a certain maximum number of iterations and a number before restart. The maximum number before restart is usually set to 10 or 20. Iterations are performed until the desired residual error is achieved or the maximum iteration count is reached. The code was modified to deliver data regardless of whether it reaches the residual error. If the residual error is not achieved, a warning on convergence will be sent to the log file for the run.

The ScaleME parameters include the number of levels and how many will be shared along with the lowest level to which the computation must be done. Setting both to one will result in a full matrix calculation and computation without using MLFMA. Usually the maximum number of levels is chosen to enhance the speed and limit the memory usage of the process. Shared levels greatly reduce the translator memory requirements and enhance the performance of parallel computations on many processors. The order of the interpolation should be at least four for accurate interpolation and anterpolation of the radiation and receiving patterns between levels. Compressed translators are used to avoid the storage and computation load on some problems. The size of the smallest box can be fixed which will generally make the top-level box larger than it needs to be to enclose the target exactly. This will have a significant impact on the memory for the translators but may reduce other memory requirements associated with the number of samples at the lowest level.

B.3.2 Stability

LSSP is not a stable code. It is still a research-level code and requires a sound understanding of MLFMA. Development within an external company would be useful to enhance the ease of use and robustness of the code. LSSP is really in its youth as far as development goes. Maturing this code will take significant testing with many targets on different platforms involving a variety of unknowns and number of processors. It is a code that deserves a great deal of respect and it will not produce satisfactory results until such respect is given.

REFERENCES

[1] N. C. Currie, Ed., *Radar Reflectivity Measurement, Techniques and Applications.* Norwood, MA: Artech House, 1989.

[2] S. Ohnuki and W. C. Chew, "Numerical accuracy of the multipole expansion 2-D MLFMA," Center for Computational Electromagnetics, University of Illinois at Urbana-Champaign, Tech. Rep. 5-02, 2002.

[3] M. L. Hastriter, S. Ohnuki, and W. C. Chew, "Error control of the translation operator in 3-D MLFMA," *Microwave Optical Technology Letters*, vol. 37, pp. 184–188, May 2003.

[4] S. Velamparambil, W. C. Chew, and M. L. Hastriter, "Scalable electromagnetic scattering calculations," in *IEEE Antennas and Propagation Symposium*, vol. 3, pp. 176–179, 2002.

[5] W. C. Chew, J.-M. Jin, E. Michielssen, and J. M. Song, Eds., *Fast and Efficient Algorithms in Computational Electromagnetics.* Boston: Artech House, 2001.

[6] V. Rokhlin, "Rapid solution of integral equation of scattering theory in two dimensions," *Journal of Computer Physics*, vol. 86, no. 6, pp. 414–439, 1990.

[7] M. L. Hastriter, S. Ohnuki, and W. C. Chew, "Error control of the translation operator in 3-D MLFMA," Center for Computational Electromagnetics, University of Illinois at Urbana-Champaign, Tech. Rep. 18-02, 2002.

[8] R. Coifman, V. Rokhlin, and S. Wandzura, "The fast multipole method for the wave equation: A pedestrian prescription," *IEEE Antennas and Propagation Magazine*, vol. 35, pp. 7–12, June 1993.

[9] J. M. Song, C. C. Lu, and W. C. Chew, "Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects," *IEEE Transactions on Antennas and Propagation*, vol. 45, pp. 1488–1493, October 1997.

[10] J. M. Song and W. C. Chew, "Fast multipole method solution using parametric geometry," *Microwave Optical Technology Letters*, vol. 7, pp. 760–765, November 1994.

[11] M. F. Gyure and M. A. Stalzer, "A prescription for the multilevel Helmholtz FMM," *IEEE Computational Science & Engineering*, vol. 5, no. 3, pp. 39–47, July-September 1998.

[12] B. Dembart and E. Yip, "The accuracy of fast multipole methods for Maxwell's equations," *IEEE Computational Science & Engineering*, vol. 5, pp. 48–56, July-September 1998.

[13] S. Koc, J.-M. Song, and W. C. Chew, "Error analysis for the numerical evaluation of the diagonal forms of the scalar spherical addition theorem," *SIAM Journal of Numerical Analysis*, vol. 36, no. 3, pp. 906–921, 1999.

[14] W. C. Chew, *Waves and Fields in Inhomogeneous Media.* New York: Van Nostrand Reinhold, 1990.

[15] J. M. Song and W. C. Chew, "Error analysis for the truncation of multipole expansion of vector Green's functions," *IEEE Microwave and Wireless Components Letters*, vol. 11, no. 7, pp. 311–313, 2001.

[16] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions.* New York: Dover Publications, 1972.

[17] S. Ohnuki and W. C. Chew, "Numerical accuracy of multipole expansion for 2D MLFMA," *IEEE Transactions on Antennas and Propagatation*, to appear.

[18] S. Ohnuki and W. C. Chew, "Truncation error analysis of multipole expansion," *SIAM J. Scientific Computing*, to appear.

[19] D. A. Dunavant, "High degree efficient symmetrical Gaussian quadrature rules for the triangle," *Int. J. Numer. Meth. Eng.*, vol. 21, pp. 1129–1148, 1985.

[20] S. Velamparambil, W. C. Chew, and J. M. Song, "10 million unknowns: Is it *that* big?" *IEEE Antennas and Propagation Magazine*, vol. 45, pp. 43–58, April 2003.

[21] S. Velamparambil and W. C. Chew, "Parallelization of multilevel fast multipole algorithm on distributed memory computers," Center for Computational Electromagnetics, University of Illinois at Urbana-Champaign, Tech. Rep. 13-01, 2001.

[22] J. M. Song and W. C. Chew, "The Fast Illinois Solver Code: Requirements and scaling properties," *IEEE Computational Science and Engineering*, vol. 5, pp. 19–23, July-September 1998.

[23] M. L. Hastriter and W. C. Chew, "Memory reduction in MLFMA through target rotation," in *IEEE Antennas and Propagation Symposium*, vol. 2, pp. 306–309, 2003.

[24] E. F. Knott, J. F. Shaeffer, and M. T. Tuley, Eds., *Radar Cross Section.* Norwood, MA: Artech House, 1993.

VITA

Michael Larkin Hastriter was born in Oregon in 1968. He received a B.S. degree in electrical engineering from Brigham Young University in 1993. He was commissioned in 1993 as a Second Lieutenant in the United States Air Force. Lieutenant Hastriter worked at Wright Laboratories, now known as Air Force Research Laboratories, at Wright-Patterson Air Force Base, Ohio. He then worked at the National Air Intelligence Center as an Aerospace Signatures Analyst. Two years later, he was given both the Air Force Achievement medal and the Air Force Commendation medal and selected to attend the Air Force Institute of Technology (AFIT).

During the 18-month AFIT program, he studied information and radar systems. He specialized in communications systems and radar technology. The results in his thesis, entitled "Range Dependent Scattering in the Intermediate-Zone," were applied to Air Force programs. He was recognized as a Distinguished Graduate (DG) from AFIT which represents the top 10% of the class. He received the M.S. degree from AFIT in 1997.

In 1997, he was assigned to the Air Force Information Warfare Center, at Kelly Air Force Base, Texas. There, he led the Advanced Signatures Section and also the Modeling and Simulation Section. He was sent to Squadron Officer School (SOS) in residence at Maxwell Air Force Base, Alabama where he was also recognized as an Air Force DG. Later, he was awarded the Military Outstanding Volunteer Service Medal for his voluntary service in the community and to the Boy Scouts of America. After SOS, he was selected to be the Signatures Branch Chief. With a budget of nearly a million dollars and a staff of officers, enlisted, civil servants, and contractor support teams, he led the effort to acquire, analyze, store, and distribute United States and allied signatures data for the entire Department of Defense. His work garnered him the Air Force Meritorious Service Medal, which is generally reserved for more senior officers.

He was then competitively selected for a teaching position at AFIT and was sent to the University of Illinois (UI) to complete a doctorate in electrical and computer engineering. During this time, Captain Hastriter was selected as a Major and identified as an intermediate service school candidate. This identification is

given to the top 20% of those meeting the Major's promotion board and provides a 70% opportunity for further professional military education, usually Air Command and Staff College. While studying at UI, he doubled the world record for solving a surface scattering problem using the multilevel fast multipole algorithm and a sphere with 20 million unknowns. He authored and published a refereed article in the *Microwave Optical Technology Letters* entitled "Error control of the translation operator in 3D MLFMA." His work has been submitted to several conferences.

Captain Hastriter will be an assistant professor in the electrical and computer engineering department of AFIT after completion of his doctorate. He is expected to pin-on the rank of Major in November 2003, the same month he and Bethany will have their seventh child.